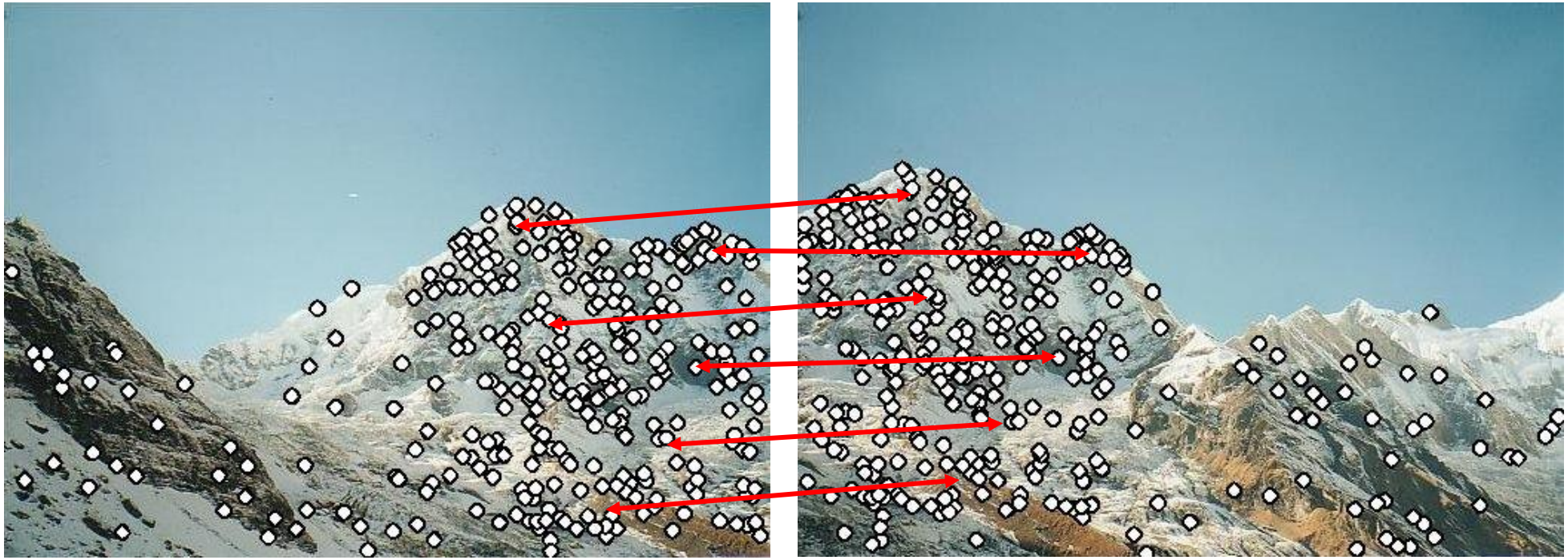
7. Feature Detector & Descriptor

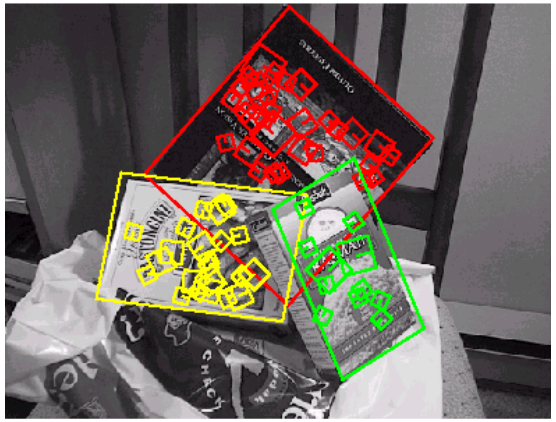


Why Extract Features?

- Planar object instance recognition

Database of planar objects

Instance recognition



Why Extract Features?

- 3D object recognition

Database of 3D objects

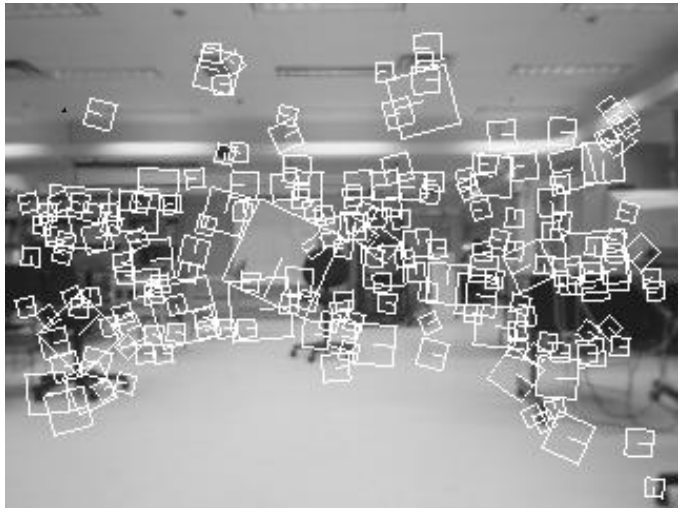
3D objects recognition

Why Extract Features?

- Location Recognition

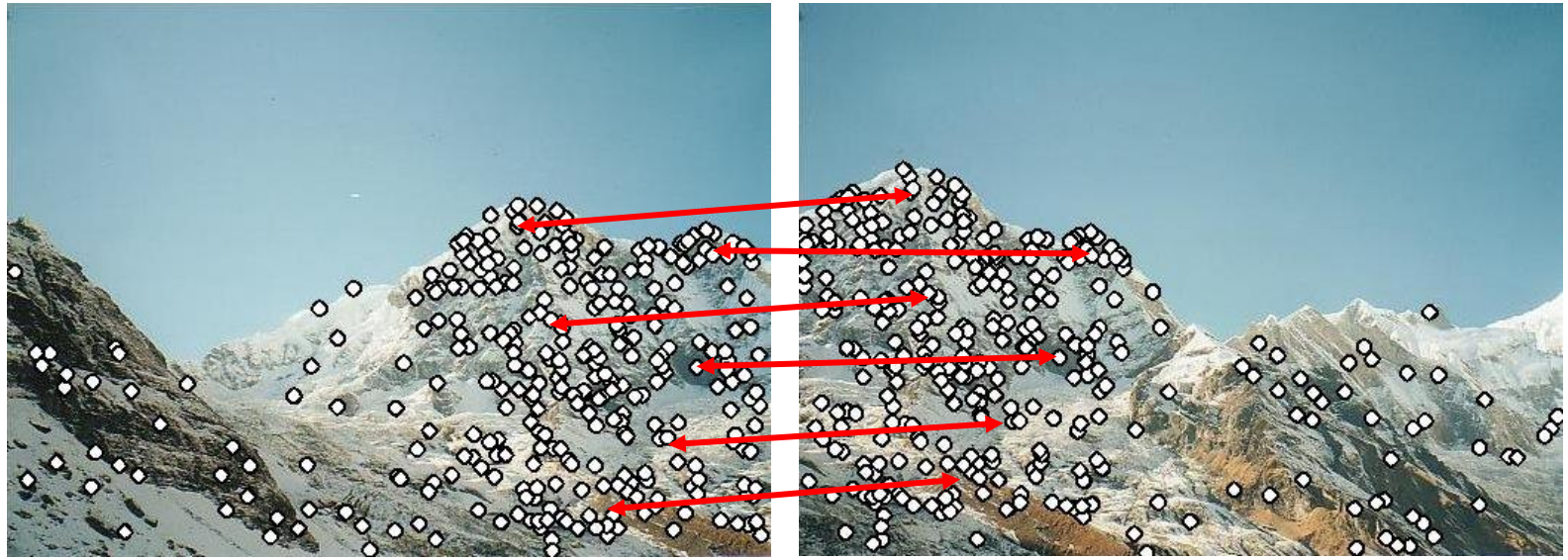
Why Extract Features?

- Robot Localization



Why Extract Features?

- Panorama image stitching



Advantages of Local Features

Locality

- features are local, so robust to occlusion and clutter

Quantity

- hundreds or thousands in a single image

Distinctiveness:

- can differentiate a large database of objects

Efficiency

- real-time performance achievable

The Feature Matching Problem

- Pick a distinctive point in an image
- Find it again in another image
- Two fundamental problems:
  - How to find a good point?
  - How to identify two points are the same?

Images are up to geometric & photometric transformations, e.g. rotation, scaling, perspective changes, exposure changes, white balance, etc.

The Feature Matching Problem

- Two problems: detection & description
- Invariant to transformations
  - geometric invariance: translation, rotation, scale
  - photometric invariance: brightness, exposure, …

Detection: What Makes a Good Feature?

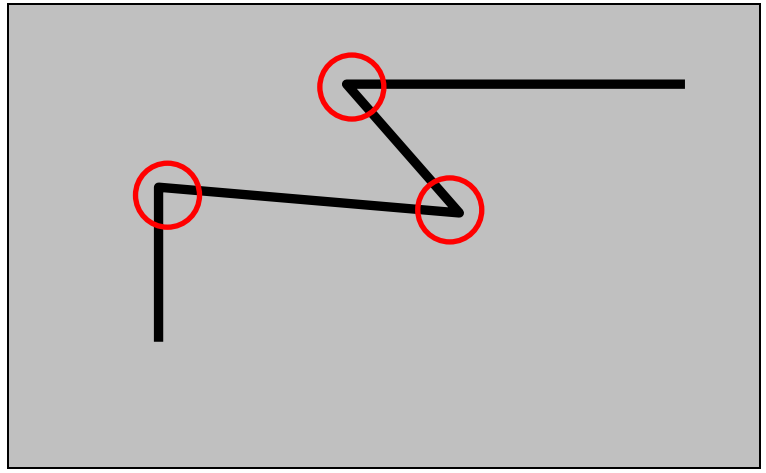
Look for image regions that are unusual

- Lead to unambiguous matches in other images

How to define “unusual”?

An introductory example:

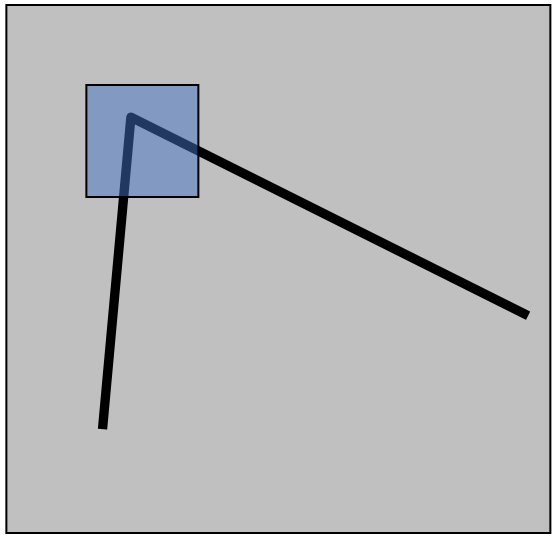
Harris corner detector



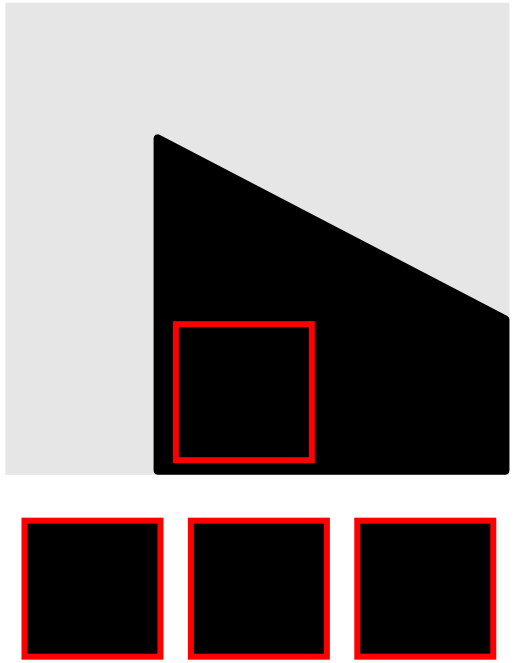
C.Harris, M.Stephens. “A Combined Corner and Edge Detector”. 1988

The Basic Idea

- Suppose we only consider a small window of pixels
  - What defines whether a feature is a good or bad candidate?
- How does the window change when you shift it?
  - We hope: shifting the window in any direction causes a big change

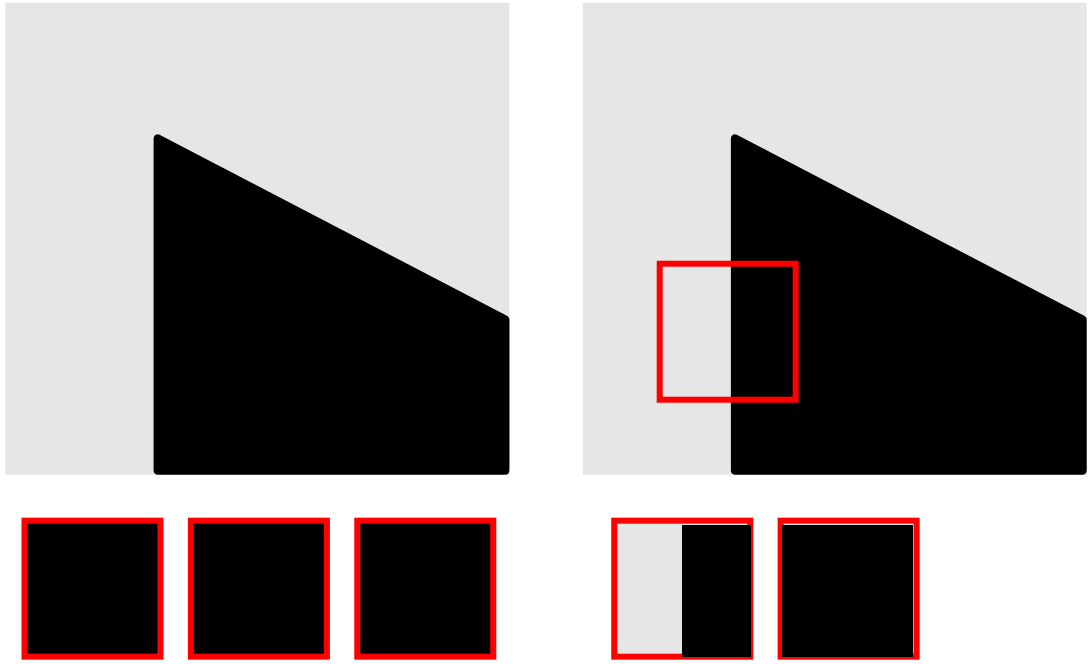


Corner Detector



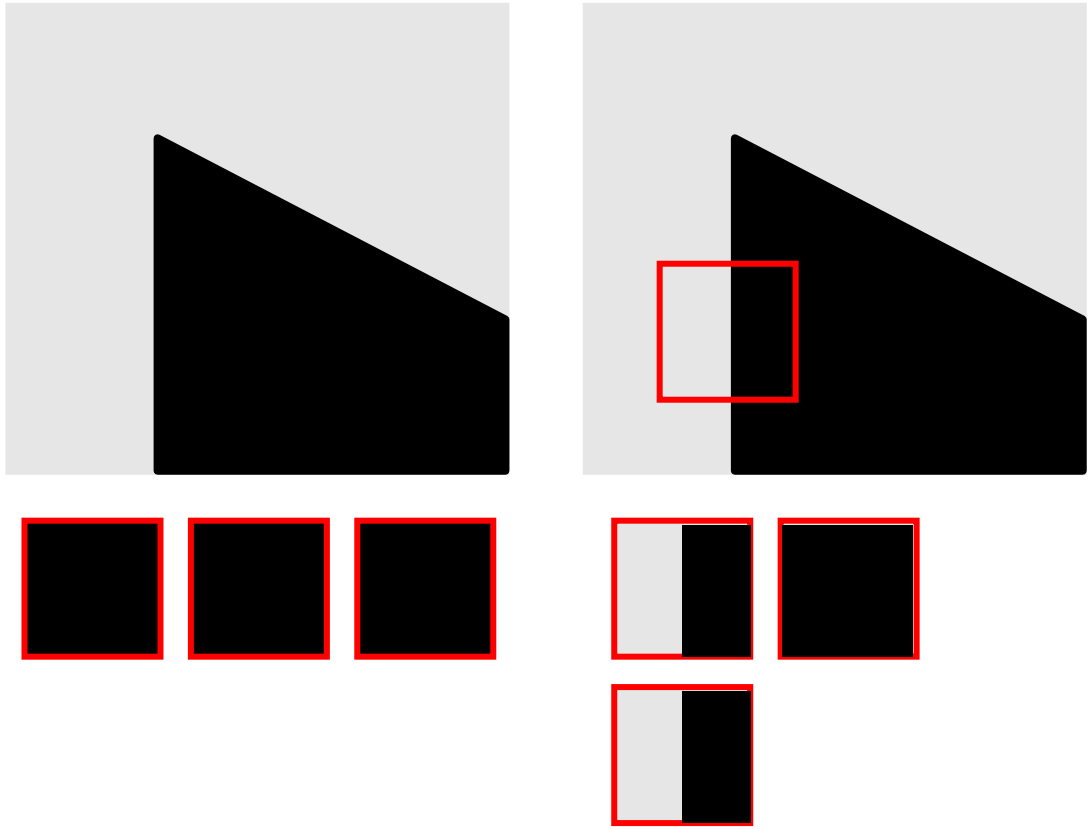
"flat" region:
no change as shift window in all directions

Corner Detector



"flat" region:
no change as shift window in all directions

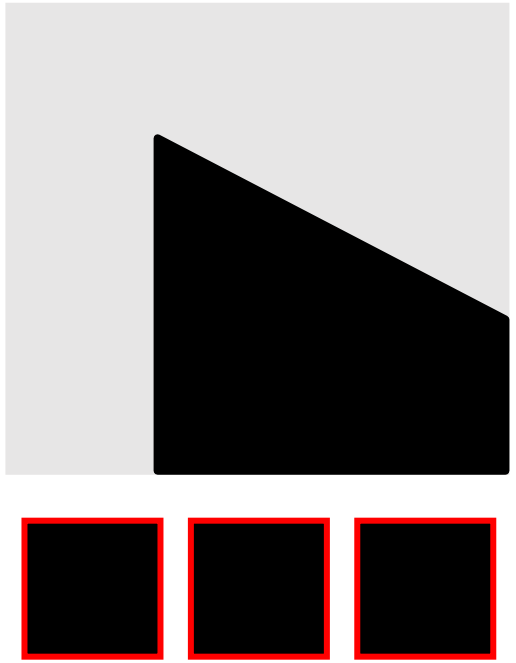
Corner Detector

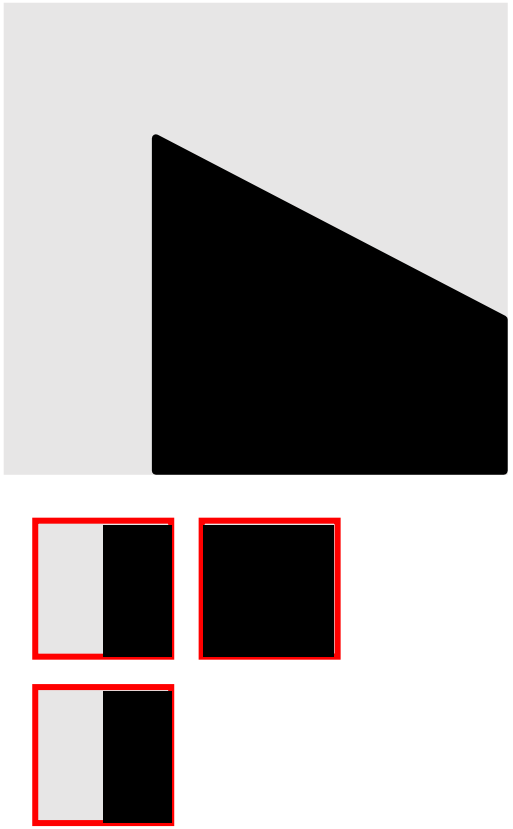


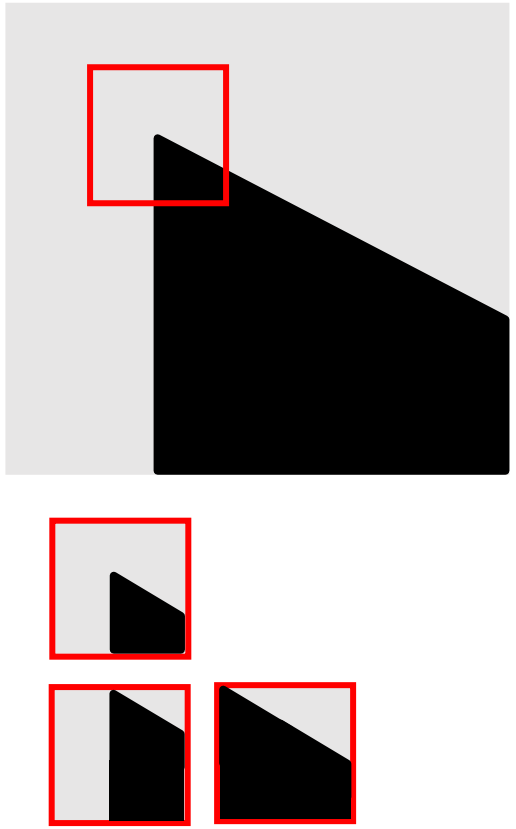
"flat" region:
no change as shift window in all directions

"edge":
no change as shift window along the edge direction

Corner Detector







"flat" region:
no change as shift window in all directions

"edge":
no change as shift window along the edge direction

"corner":
significant change as shift window in all directions

Questions?

Harris Detector: Mathematics

Shift the window by $[u,v]$:

- How do pixel values change?
- Compare each pixel before and after shift by summing up the squared differences (SSD)
- This SSD is defined as:

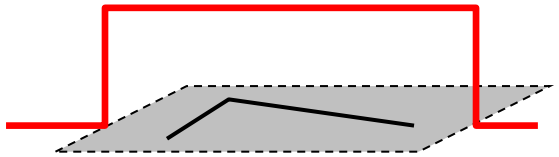
$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

weights

Shifted intensity

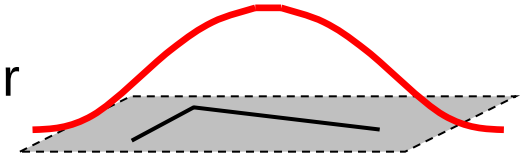
Intensity

weights $w(x,y)$ =



1 in window, 0 outside

or



Gaussian

Taylor Series Approx to Shifted Image

- When the motion $[u,v]$ is small
- Taylor expansion of image intensity I

$$E(u,v) \approx \sum_{x,y} w(x,y)[I(x,y) + uI_x + vI_y - I(x,y)]^2$$

$$= \sum_{x,y} w(x,y)[uI_x + vI_y]^2$$

$$= (u \quad v)\begin{pmatrix} A & B \\ B & C \end{pmatrix}\begin{pmatrix} u \\ v \end{pmatrix}$$

$$A = \sum_{x,y} w(x,y) I_x^2, \quad B = \sum_{x,y} w(x,y) I_x \, I_y, \quad C = \sum_{x,y} w(x,y) I_y^2$$

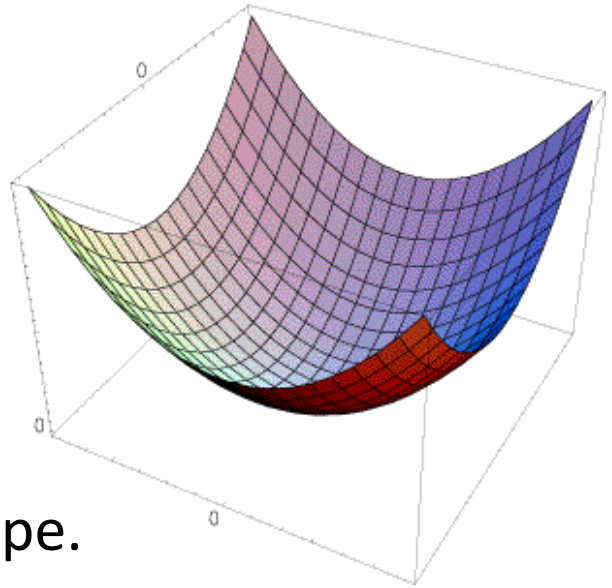
Harris Detector: Mathematics

- We have, for small shifts $[u,v]$, a *bilinear* approximation:

$$E(u,v) \cong \begin{bmatrix} u, v \end{bmatrix} \quad M \quad \begin{bmatrix} u \\ v \end{bmatrix}$$

where M is a 2×2 matrix computed from image derivatives:

$$M = \begin{pmatrix} A & B \\ B & C \end{pmatrix}$$



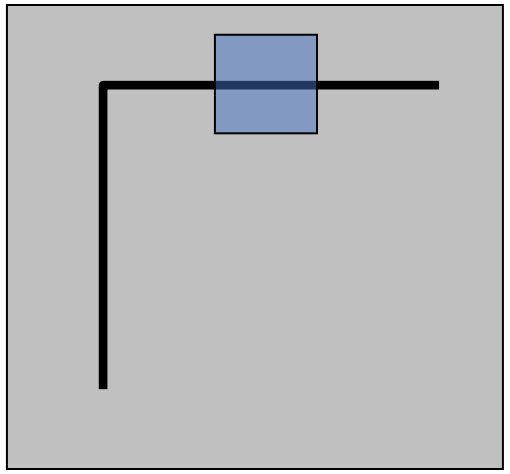
Let's try to understand its shape.

$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} \underbrace{\begin{bmatrix} A & B \\ B & C \end{bmatrix}}_{M} \begin{bmatrix} u \\ v \end{bmatrix}$$

$$A = \sum_{(x,y)\in W} I_x^2$$

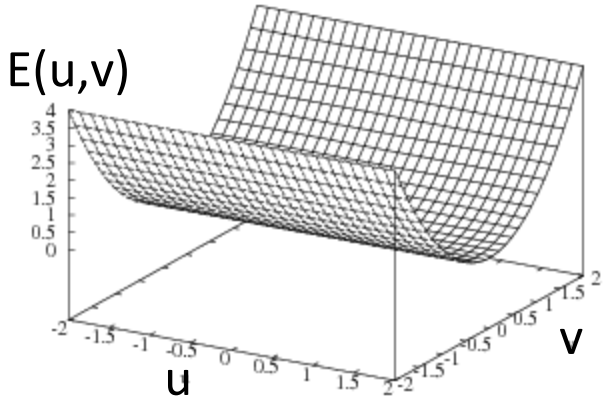
$$B = \sum_{(x,y)\in W} I_x I_y$$

$$C = \sum_{(x,y)\in W} I_y^2$$



Horizontal edge: $I_x = 0$

$$M = \begin{bmatrix} 0 & 0 \\ 0 & C \end{bmatrix}$$

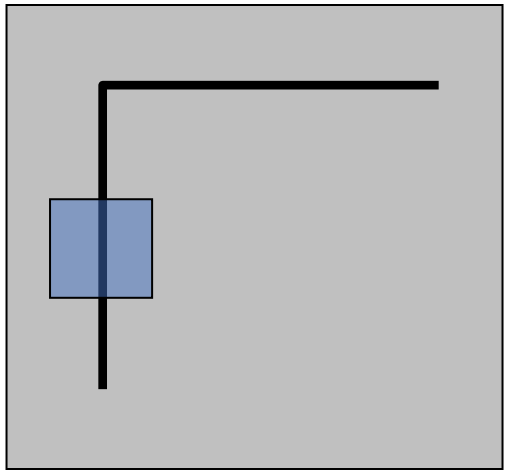


$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} \underbrace{\begin{bmatrix} A & B \\ B & C \end{bmatrix}}_{M} \begin{bmatrix} u \\ v \end{bmatrix}$$

$$A = \sum_{(x,y)\in W} I_x^2$$

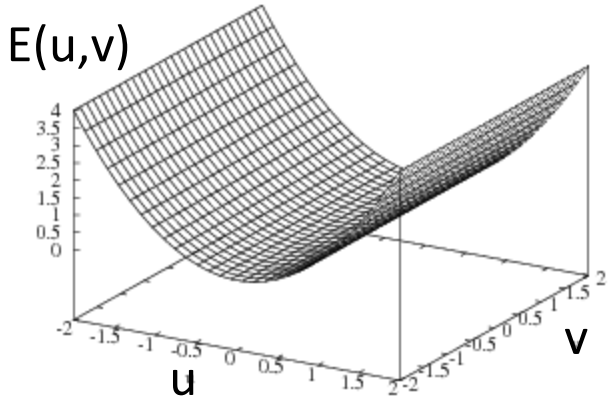
$$B = \sum_{(x,y)\in W} I_x I_y$$

$$C = \sum_{(x,y)\in W} I_y^2$$



Vertical edge: $I_y = 0$

$$M = \begin{bmatrix} A & 0 \\ 0 & 0 \end{bmatrix}$$

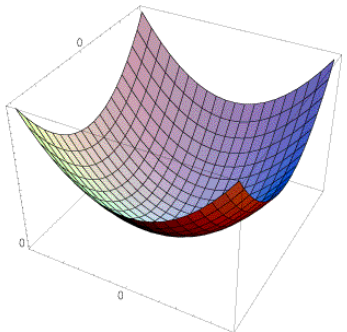


General Case

- The shape of M tells us something about the *distribution of gradients* around a pixel
- We can visualize M as an ellipse with axis lengths determined by the *eigenvalues* of M and orientation determined by the *eigenvectors* of M

$$E(u,v) \cong [u,v] \; M \begin{bmatrix} u \\ v \end{bmatrix}$$

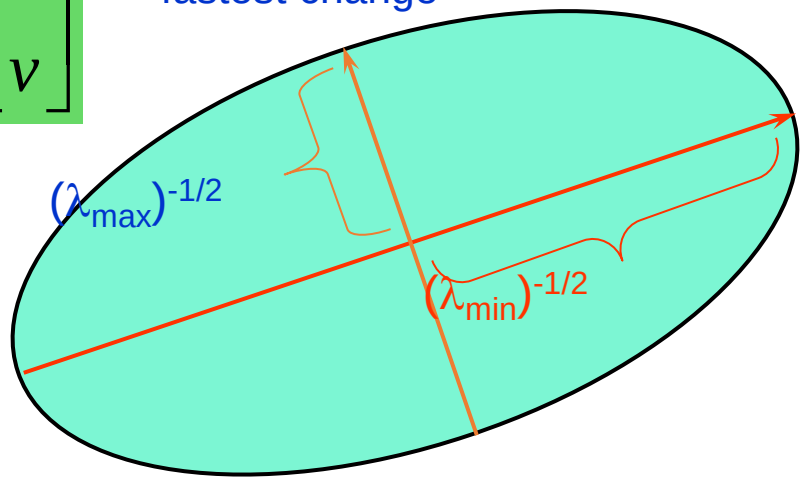
Ellipse $E(u,v)$ = const



direction of the fastest change

$\lambda_{max}, \lambda_{min}$ – eigenvalues of *M*

direction of the slowest change



quick eigenvalue/eigenvector review

The **eigenvectors** of a matrix **A** are the vectors **x** that satisfy:

$$Ax = \lambda x$$

The scalar λ is the **eigenvalue** corresponding to **x**

- The eigenvalues are found by solving:

$$det(A - \lambda I) = 0$$

- In our case, ***A*** = ***H*** is a 2x2 matrix, so we have

$$det \begin{bmatrix} h_{11} - \lambda & h_{12} \\ h_{21} & h_{22} - \lambda \end{bmatrix} = 0$$

- The solution:

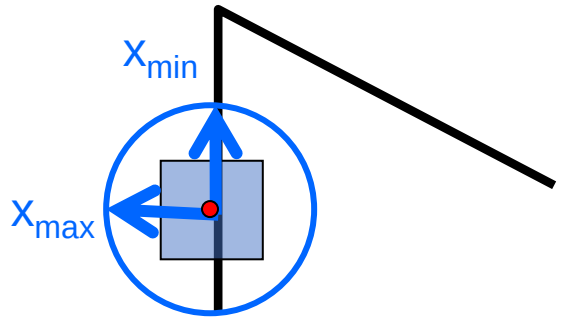
$$\lambda_{\pm} = \frac{1}{2} \left[(h_{11} + h_{22}) \pm \sqrt{4h_{12}h_{21} + (h_{11} - h_{22})^2} \right]$$

Once you know λ, you find **x** by solving

$$\begin{bmatrix} h_{11} - \lambda & h_{12} \\ h_{21} & h_{22} - \lambda \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 0$$

Harris Detector: Mathematics

$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} \underbrace{\begin{bmatrix} A & B \\ B & C \end{bmatrix}}_{M} \begin{bmatrix} u \\ v \end{bmatrix}$$



$$Mx_{max} = \lambda_{max} x_{max}$$
$$Mx_{min} = \lambda_{min} x_{min}$$

Eigenvalues and eigenvectors of M

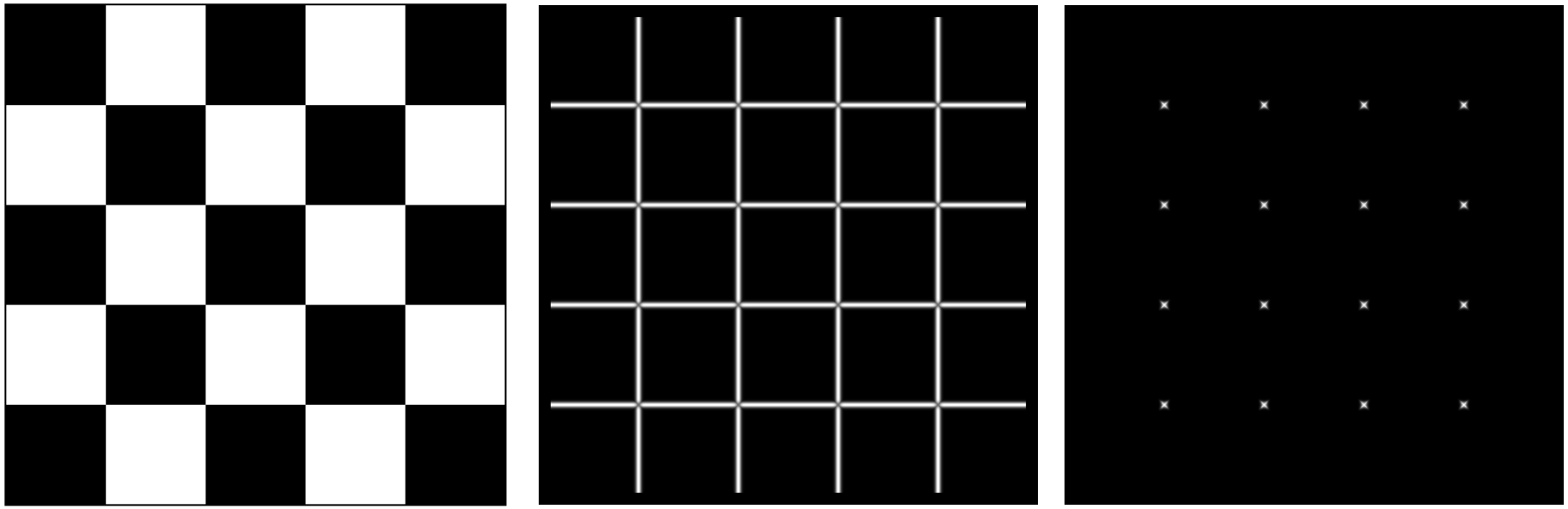
- Define shift directions with the smallest and largest change in error
- x_{max} = direction of largest increase in E
- λ_{max}= amount of increase in direction x_{max}
- $x_{\min}$ = direction of smallest increase in E
- $\lambda_{\min}$= amount of increase in direction $x_{\min}$

Harris Detector: Mathematics

Want $E(u,v)$ to be large for small shifts in all directions

the minimum of $E(u,v)$ should be large, over all unit vectors $[u,v]$

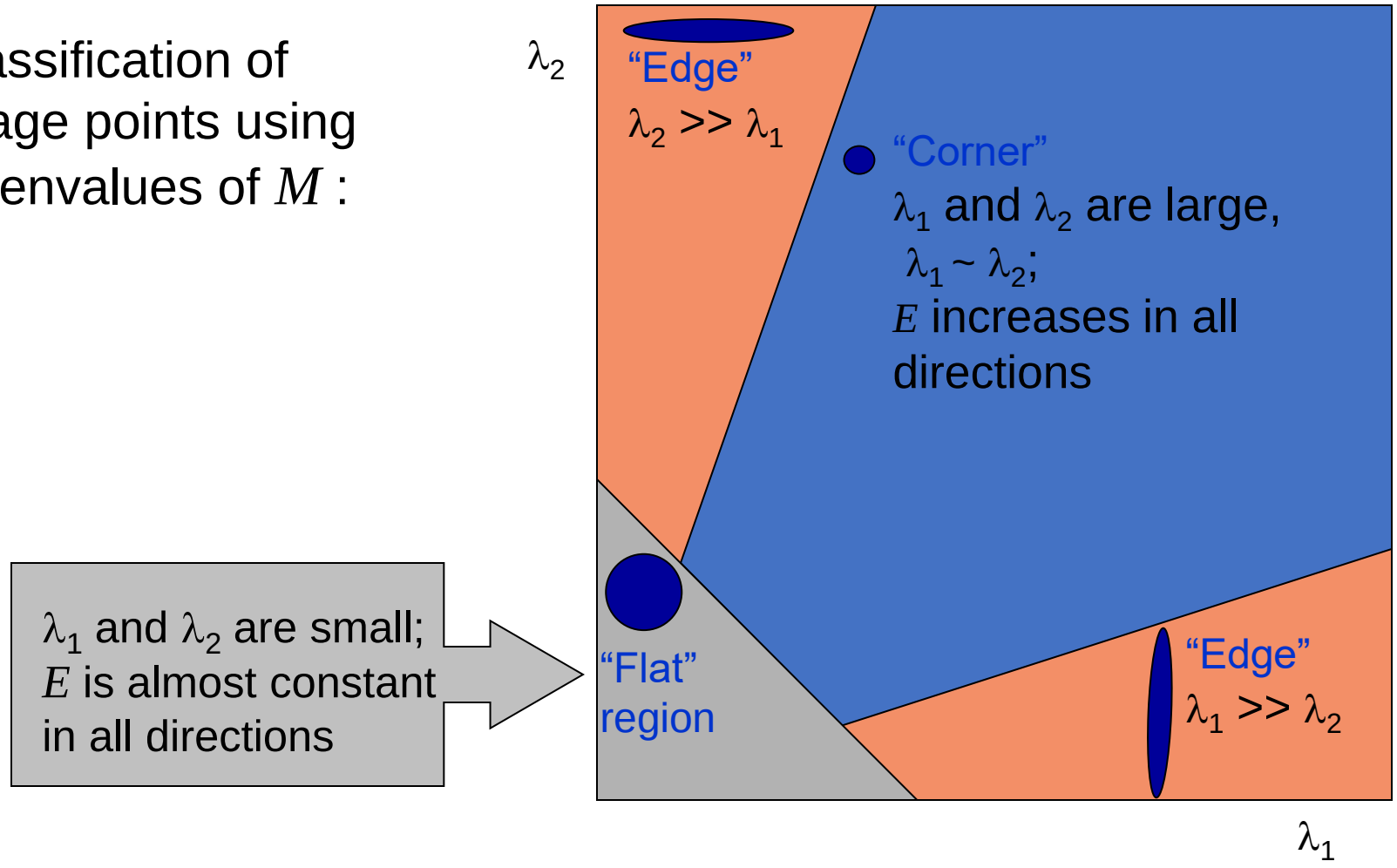
this minimum is given by the smaller eigenvalue (λ_{min}) of M



I $\lambda_{\max}$ $\lambda_{\min}$

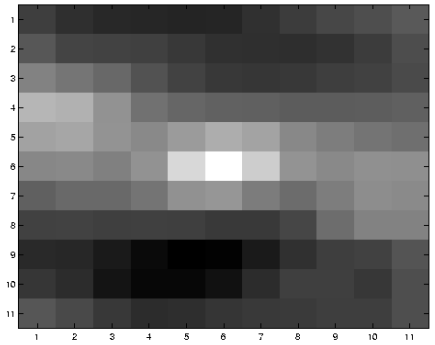
Harris Detector: Mathematics

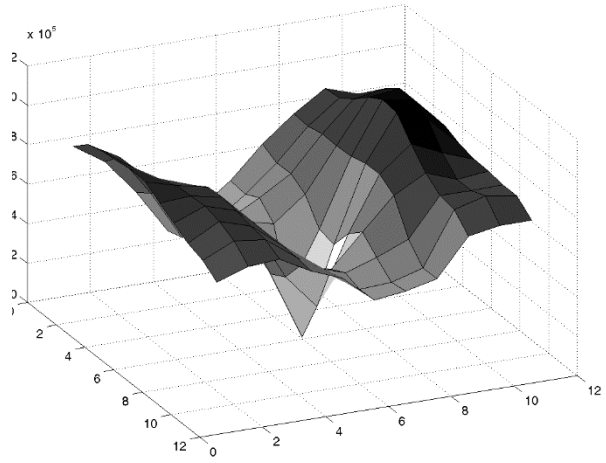
Classification of image points using eigenvalues of M :



Questions?

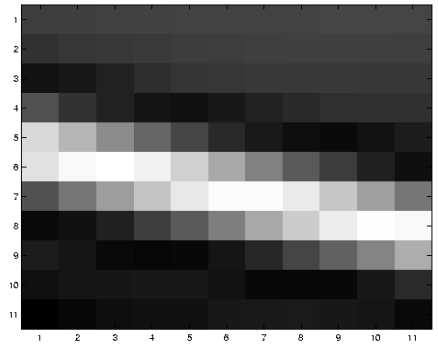
Selecting Good Features

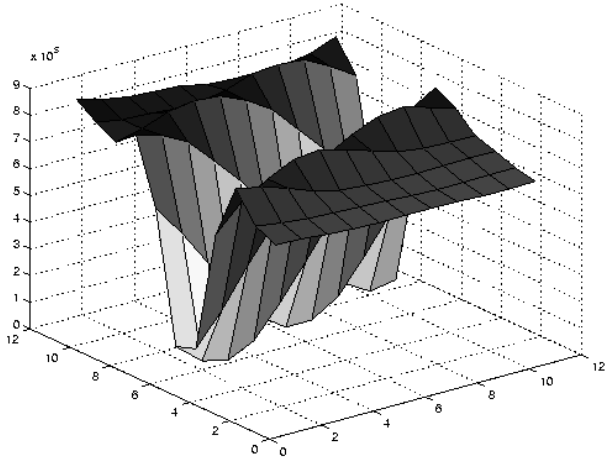




λ_1 and λ_2 are large

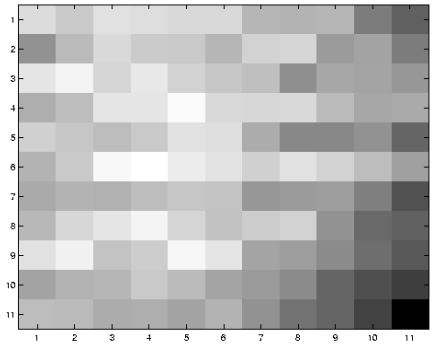
Selecting Good Features

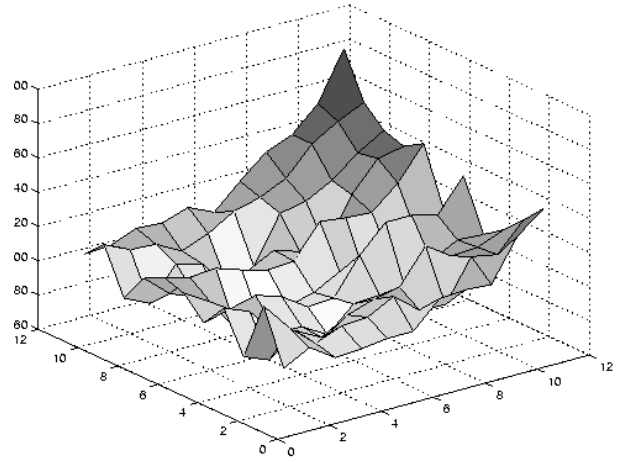




large λ_1, small λ_2

Selecting Good Features





small λ_1, small λ_2

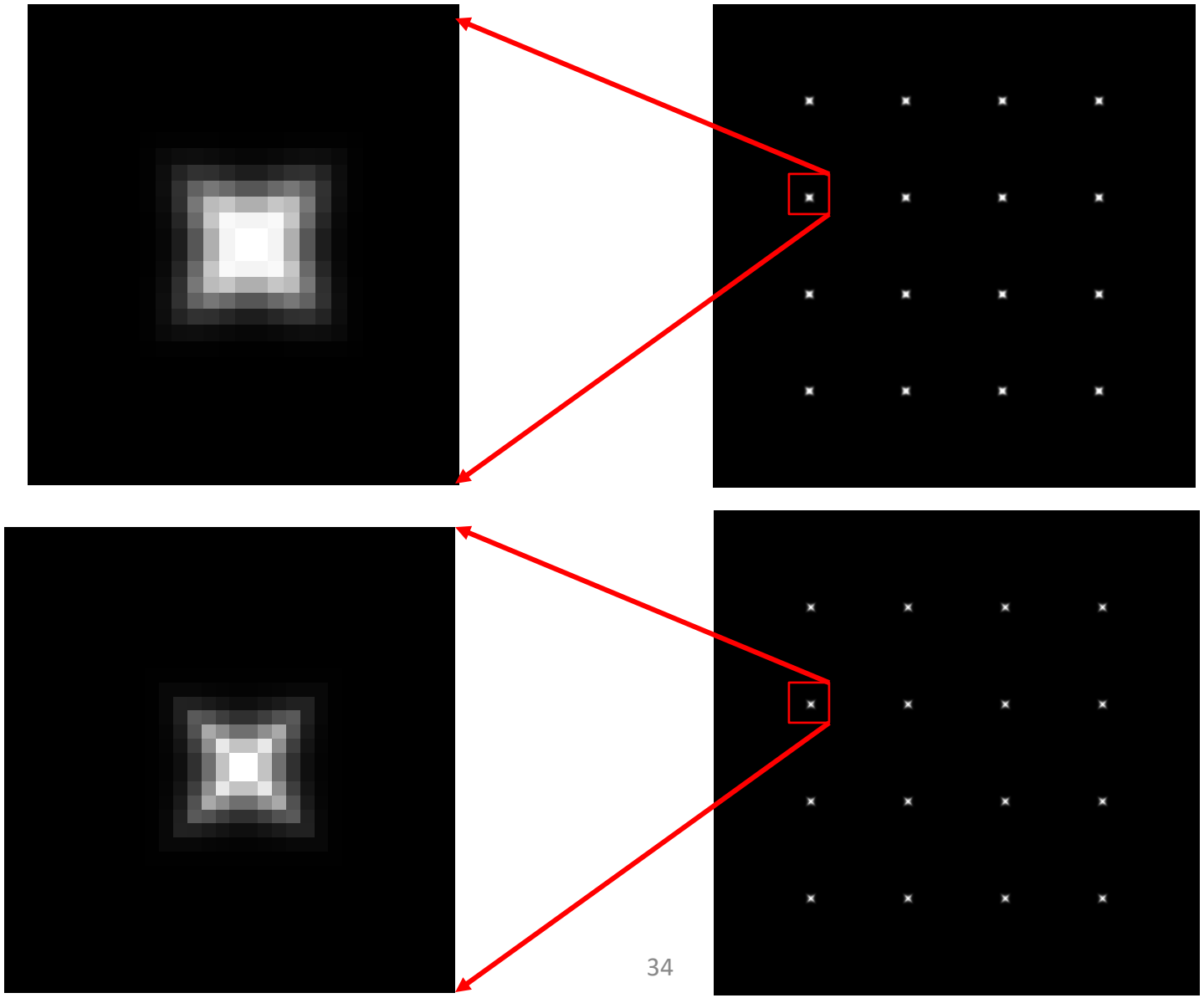
The Harris Operator

- λ_{min} is a variant of the "Harris operator" for feature detection

$$f = \frac{\lambda_{max}\lambda_{min}}{\lambda_{max} + \lambda_{min}} = \frac{\det(M)}{trace(M)}$$

- The $trace$ is the sum of the diagonals,
  - i.e., $trace(M) = h_{11} + h_{22}$
- Very similar to λ_{min} but less computationally expensive
- Called the "Harris Corner Detector" or "Harris Operator"
- Lots of other detectors, this is one of the most popular

The Harris Operator



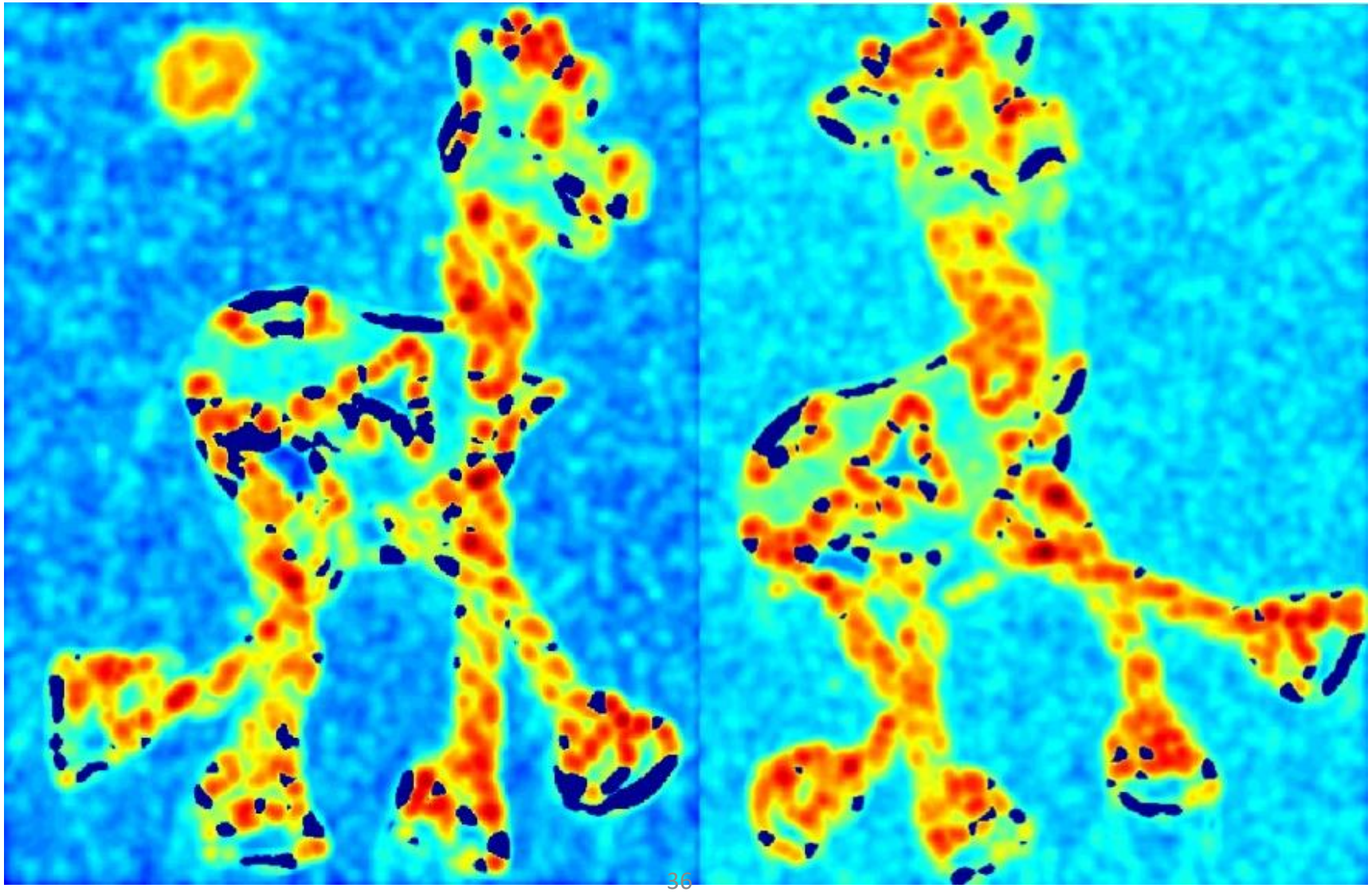
Harris operator

$\lambda_{\min}$

Harris Detector: Workflow

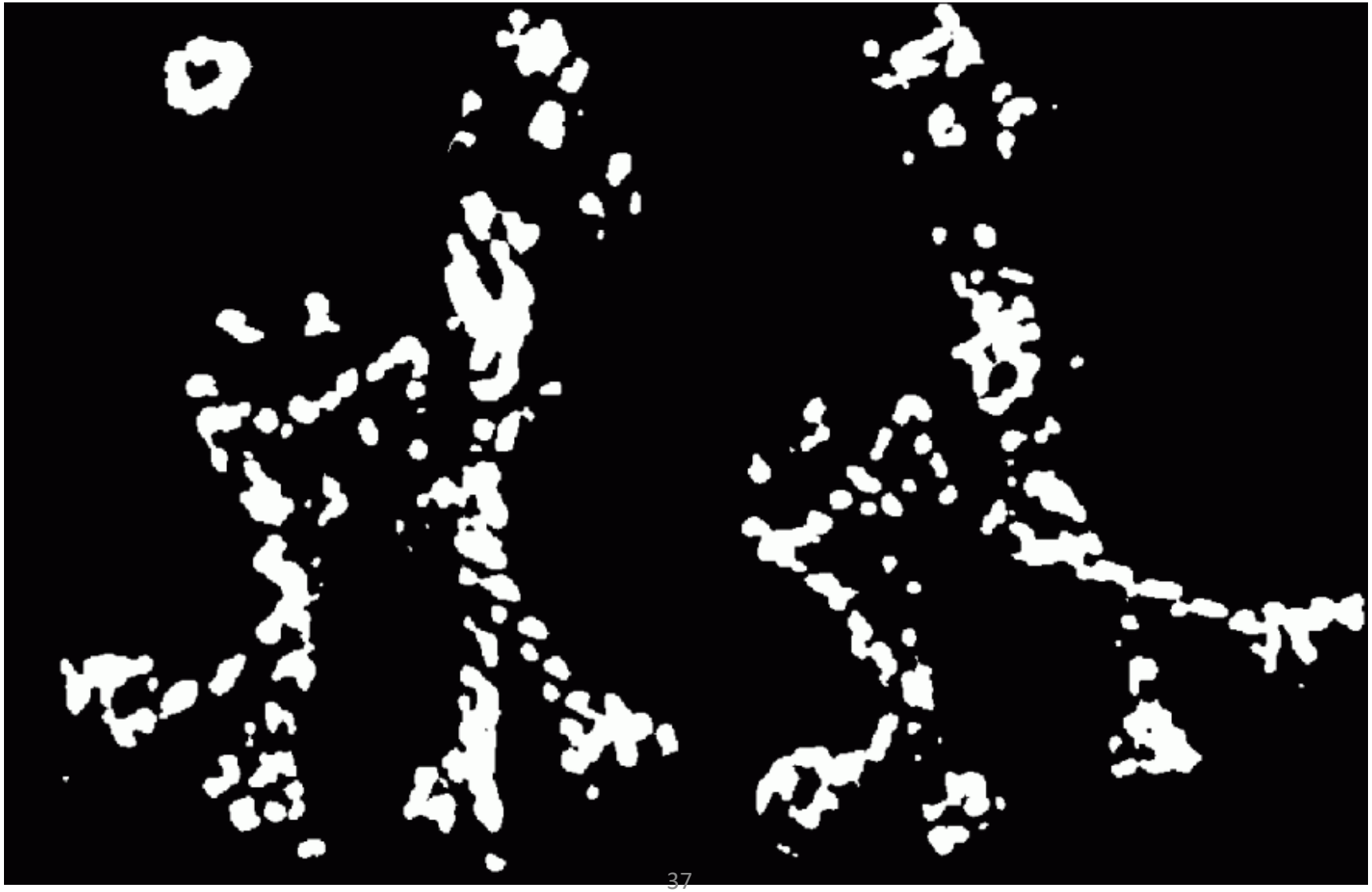
Harris Detector: Workflow

Compute corner response f



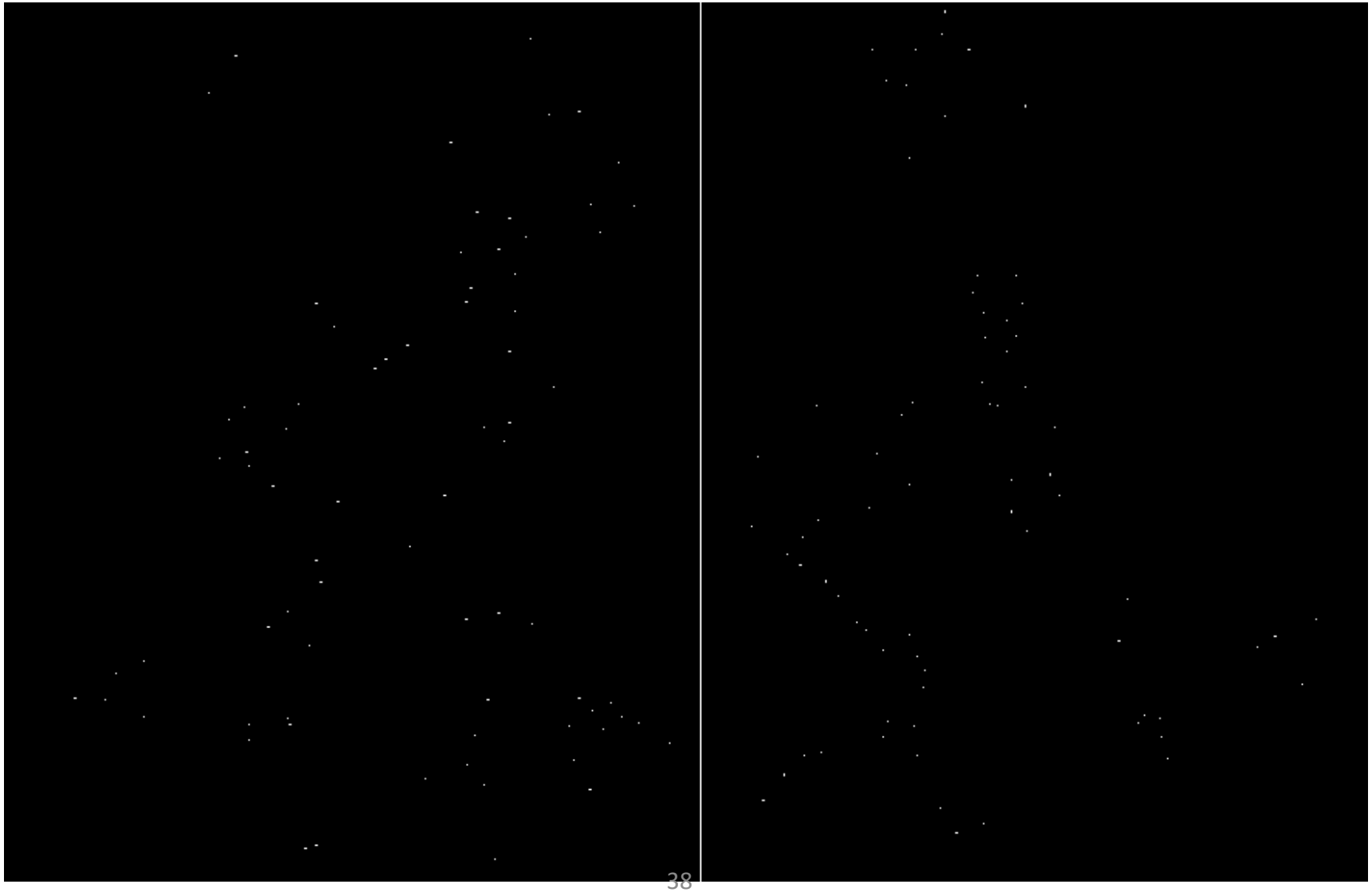
Harris Detector: Workflow

Find points with large corner response: f >threshold



Harris Detector: Workflow

Take only the points of local maxima of f (non-maximum suppression)



Harris Detector: Workflow

Harris Detector: Summary

- Average intensity change in direction $[u,v]$ can be expressed as a bilinear form:

$$E(u,v) \cong \begin{bmatrix} u,v \end{bmatrix} \; M \; \begin{bmatrix} u \\ v \end{bmatrix}$$

- Describe a point in terms of eigenvalues of M:
  measure of corner response

$$f = \frac{\lambda_{max}\lambda_{min}}{\lambda_{max} + \lambda_{min}} = \frac{\det(M)}{trace(M)}$$

- A good (corner) point should have a *large intensity change* in *all directions,*
  i.e. f should be large

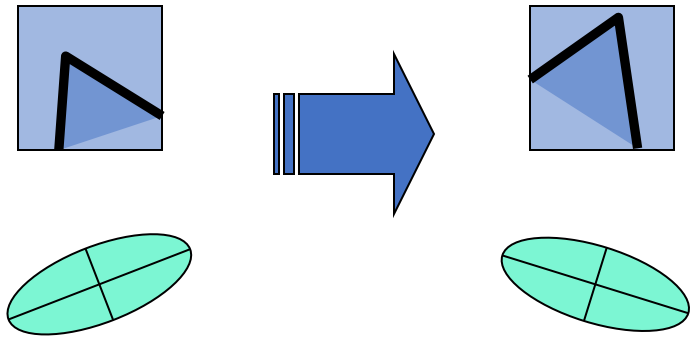
Questions?

Invariance

- Suppose you **rotate** the image by some angle
  Will you still pick up the same features?
- What if you change the brightness?
- Scale?

Harris Detector: Some Properties

- Rotation invariance



Ellipse (iso-contour of the function E) rotates but its shape (i.e. eigenvalues) remains the same. Why? Think about the meaning of E.

Corner response f is invariant to image rotation

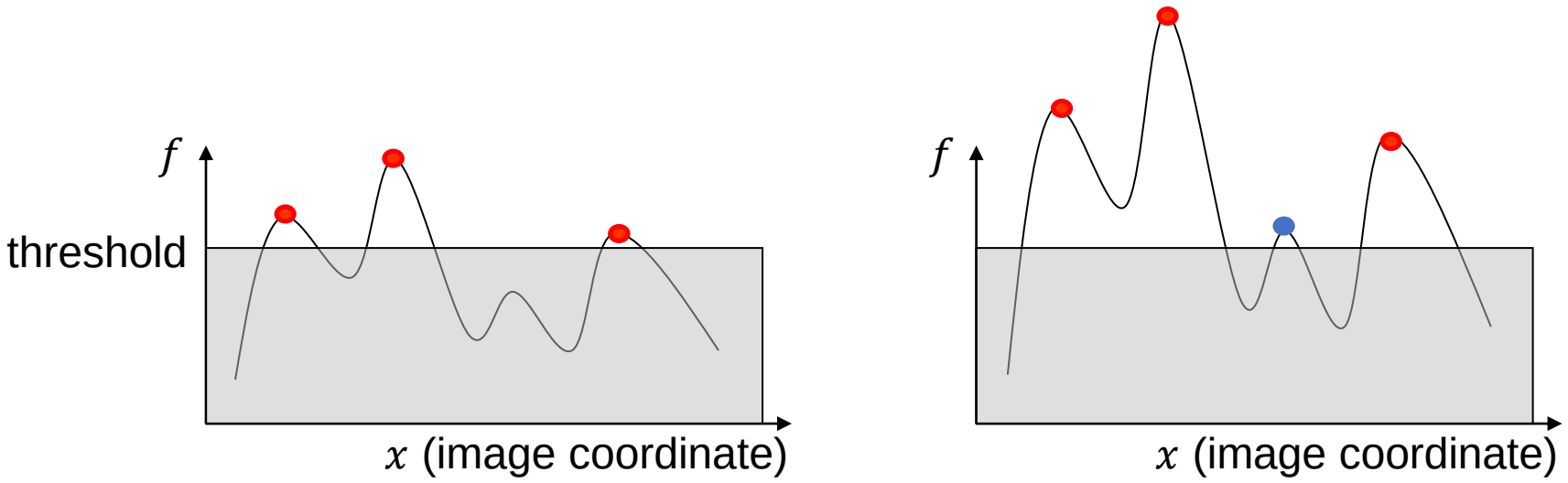
Harris Detector: Some Properties

- Partial invariance to additive and multiplicative intensity changes

  ✓ Only derivatives are used => invariance to intensity shift $I \rightarrow I + b$

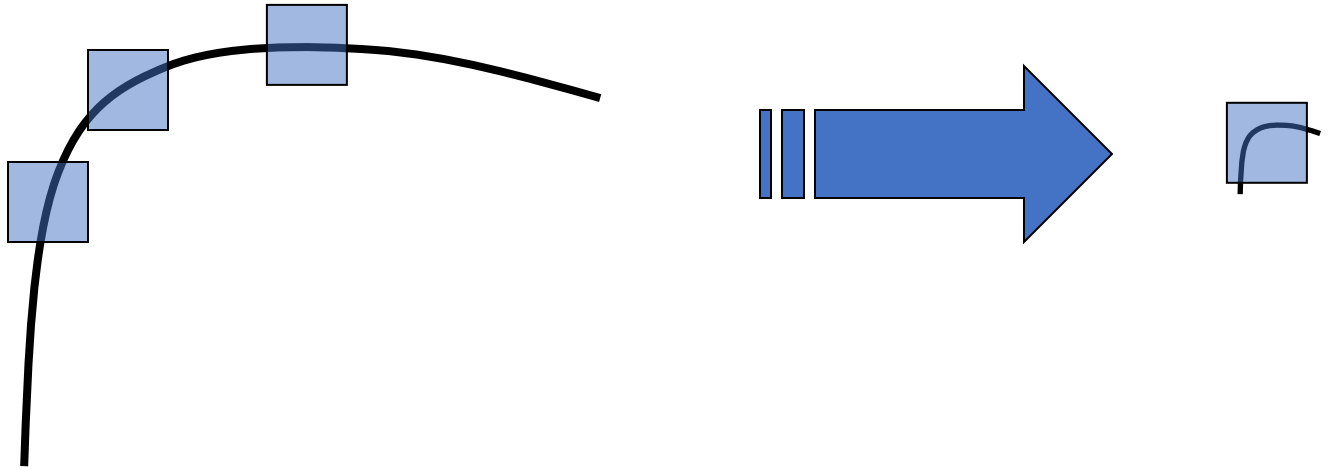
  ✓ Intensity scale: $I \rightarrow a\, I$

  Because of fixed intensity threshold on local maxima, only partial invariance to multiplicative intensity changes.



Harris Detector: Some Properties

- Not invariant to *image scale*!



All points will be classified as edges

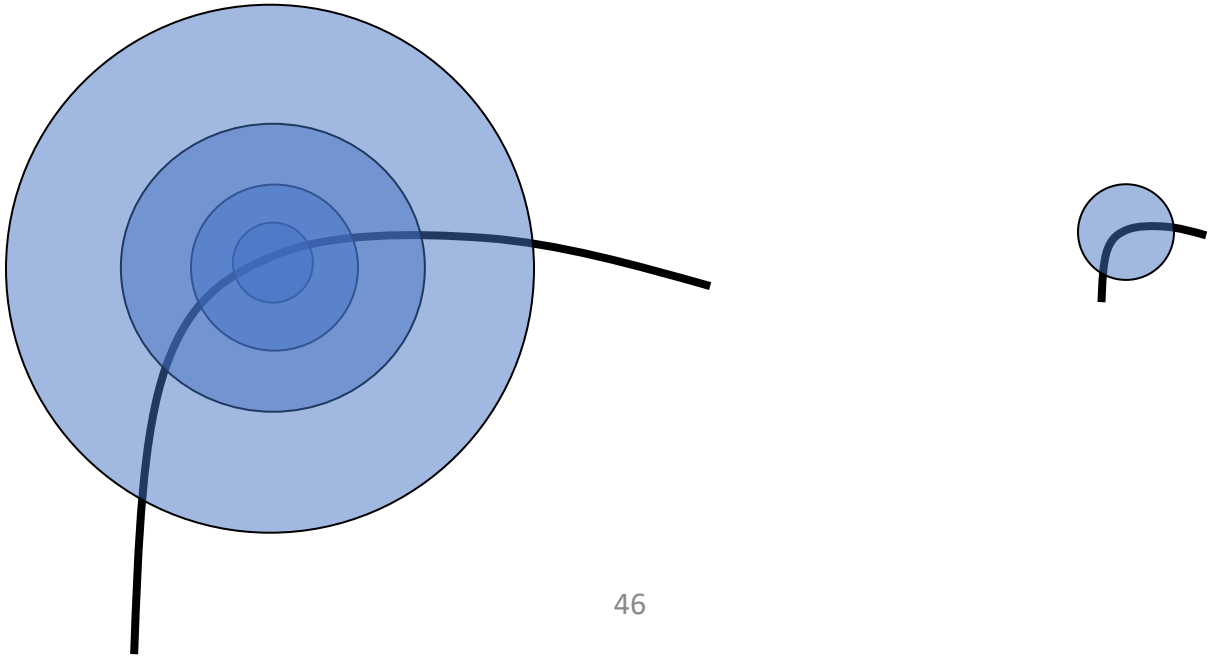
Corner !

Scale Invariant Detection

Suppose you are looking for corners

Key idea: find scale that gives local maximum of f

- Search in both position and scale
- One choice of f: the Harris operator



Evaluation of Feature Detectors

International Journal of Computer Vision 37(2), 151–172, 2000

Evaluation of Interest Point Detectors

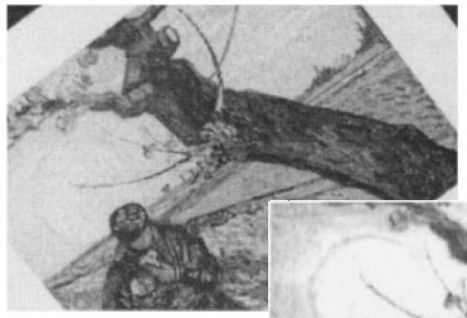
CORDELIA SCHMID, ROGER MOHR AND CHRISTIAN BAUCKHAGE
INRIA Rhône-Alpes, 655 av. de l'Europe, 38330 Montbonnot, France
Cordelia.Schmid@inrialpes.fr

Abstract. Many different low-level feature detectors exist and it is widely agreed that the evaluation of detectors is important. In this paper we introduce two evaluation criteria for interest points: repeatability rate and information content. Repeatability rate evaluates the geometric stability under different transformations. Information content measures the distinctiveness of features. Different interest point detectors are compared using these two criteria. We determine which detector gives the best results and show that it satisfies the criteria well.

Evaluation of Detectors

- Experimental setting:
  - Capture images of a planar scene with different rotation, zooming, noise and lighting configuration
  - Obtain perfect registration between these images separately
  - See if the same feature point can be detected in both images (within certain distance)
  - Quality is measured by

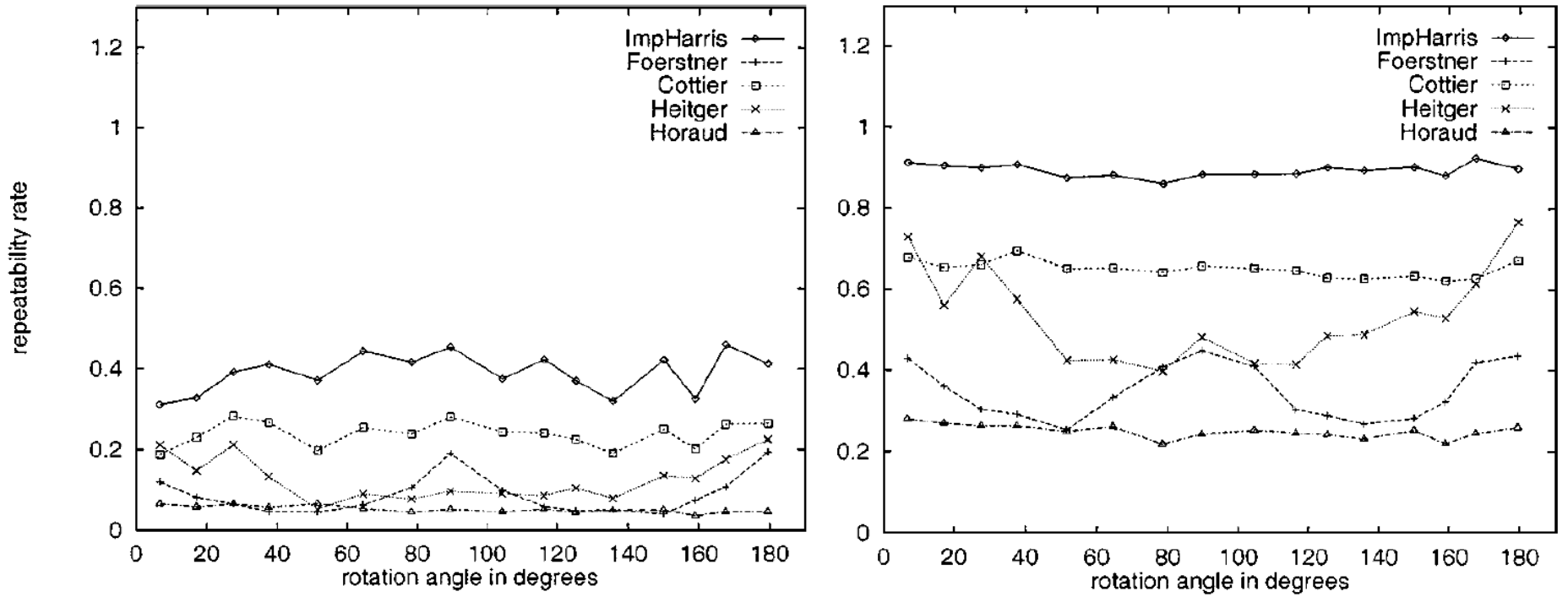
Repeatability rate = $\dfrac{\text{\# correspondences}}{\text{\# possible correspondences}}$



$$r_i(\epsilon) = \frac{|R_i(\epsilon)|}{\min(n_1, n_i)} \quad R_i(\epsilon) = \{(\tilde{x}_1, \tilde{x}_i) \mid dist(H_{1i}\tilde{x}_1, \tilde{x}_i) < \epsilon\}$$

Rotation Invariant Detection

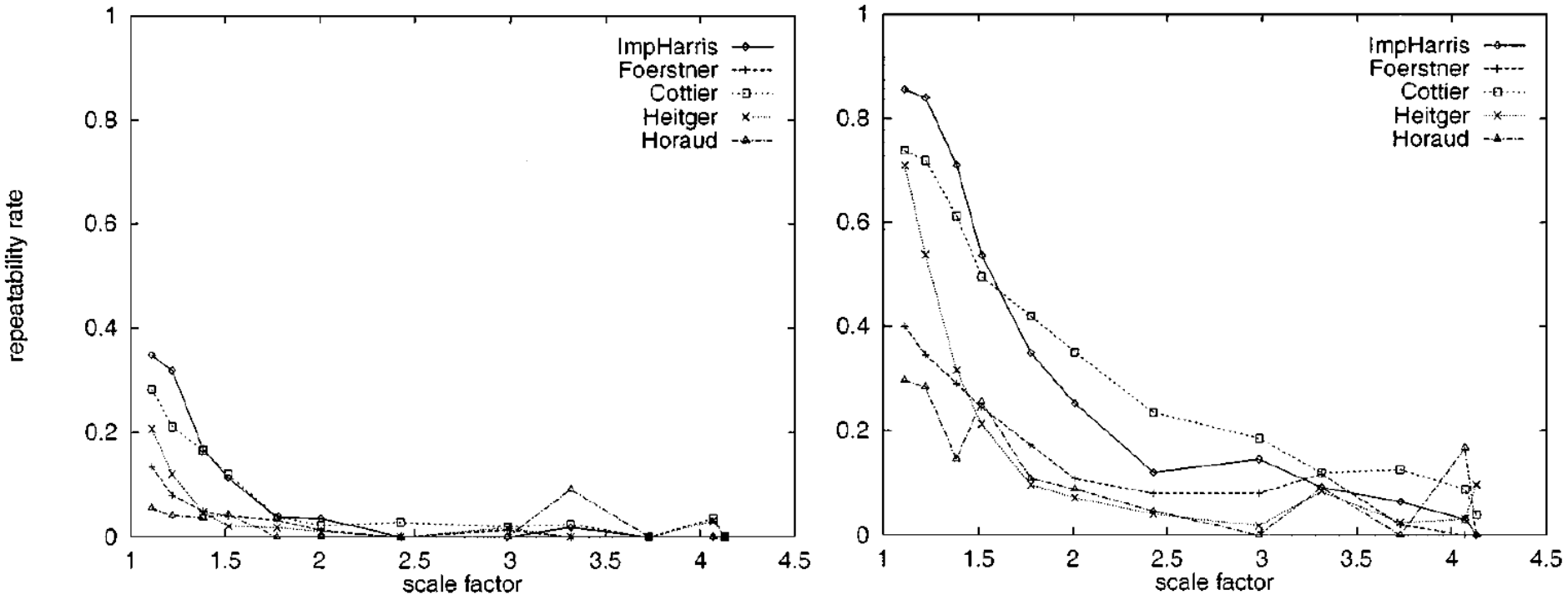
- Harris detector is invariant to rotations
  - The left and right are results when the threshold is 0.5 or 1.5 pixels
  - Note even when the rotation is 0, the repeatability ratio is not 1



C.Schmid et.al. “Evaluation of Interest Point Detectors”. IJCV 2000

It is NOT scale invariant!

- Quality of Harris detector for different scale changes

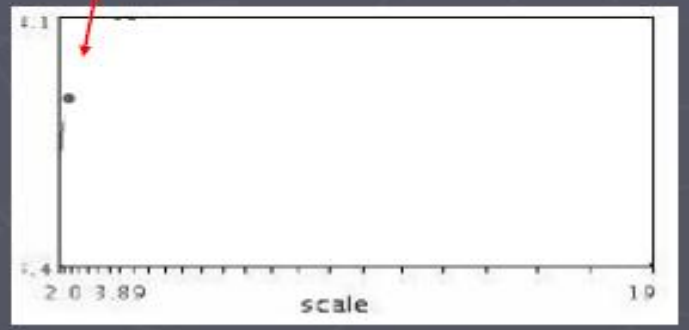


C.Schmid et.al. “Evaluation of Interest Point Detectors”. IJCV 2000

Questions?

Automatic scale selection

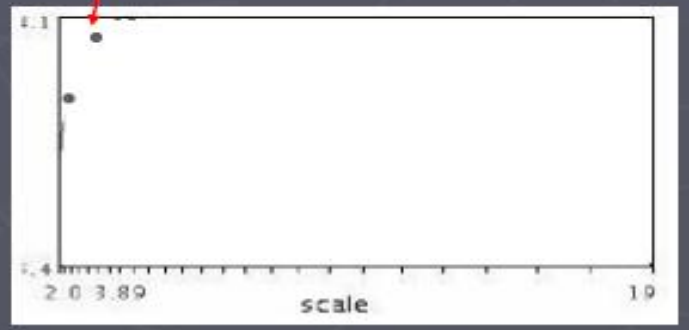
Lindeberg et al., 1996



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

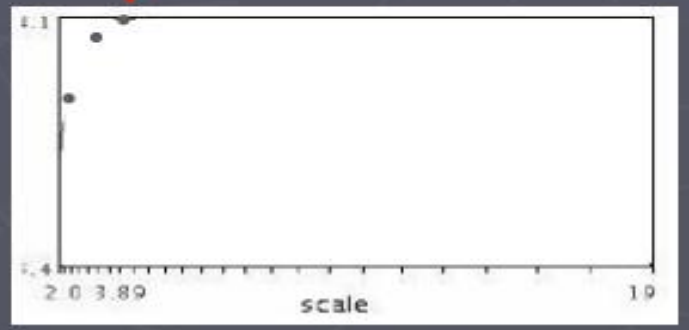
Slide from Tinne Tuytelaars

Automatic scale selection



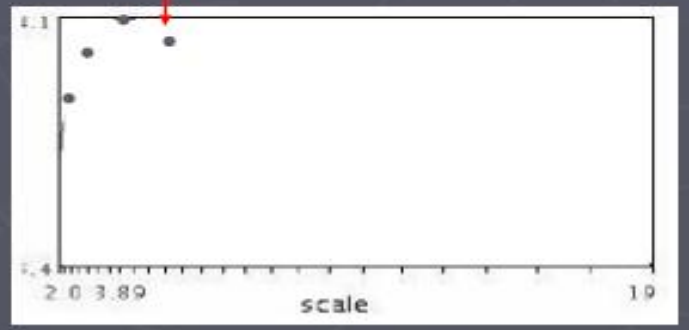
$$f(I_{i_1 \cdots i_m}(x,\sigma))$$

Automatic scale selection



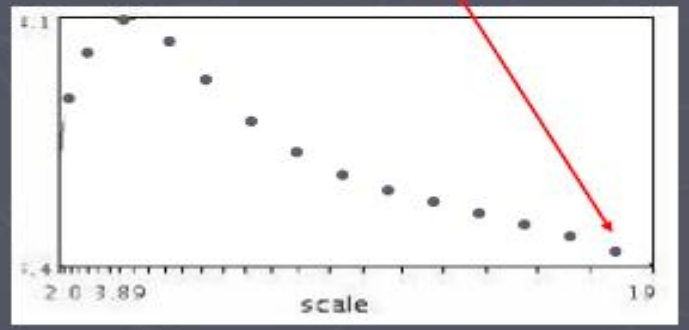
$$f(I_{i_1\cdots i_m}(x,\sigma))$$

Automatic scale selection



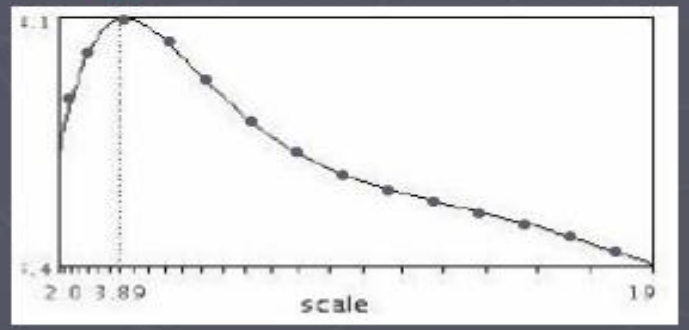
$$f(I_{i_1 \ldots i_m}(x,\sigma))$$

Automatic scale selection



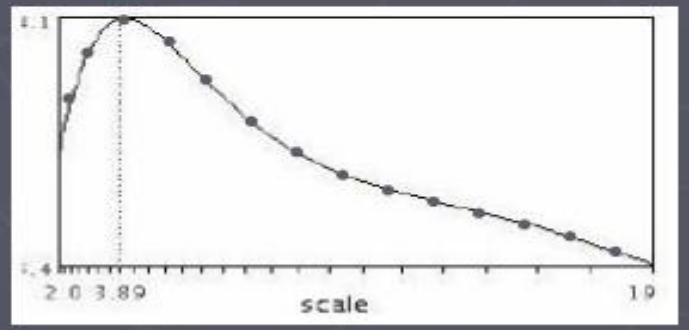
$$f(I_{i_1 \ldots i_m}(x,\sigma))$$

Automatic scale selection

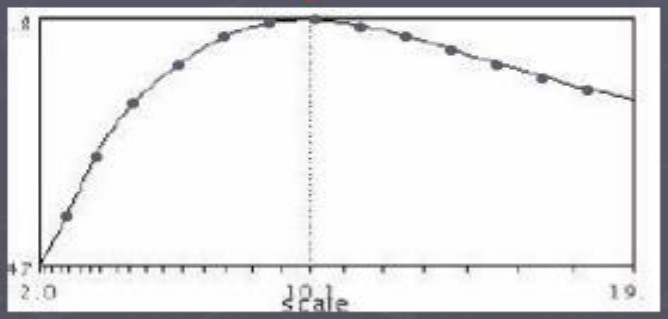


$$f(I_{i_1 \cdots i_m}(x,\sigma))$$

Automatic scale selection



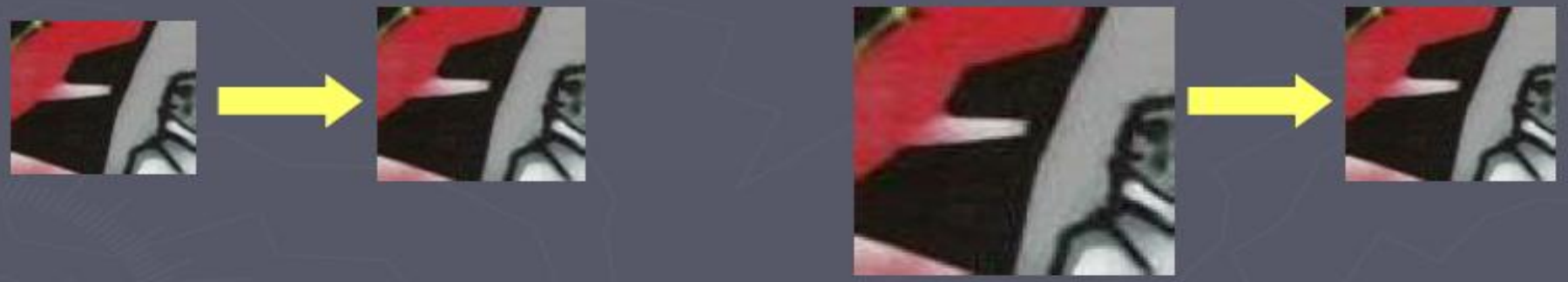
$f(I_{i_1 \ldots i_m}(x,\sigma))$



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

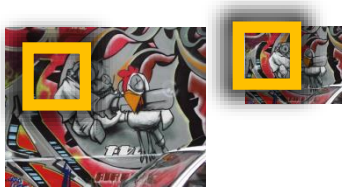
Automatic scale selection

Normalize: rescale to fixed size



Implementation

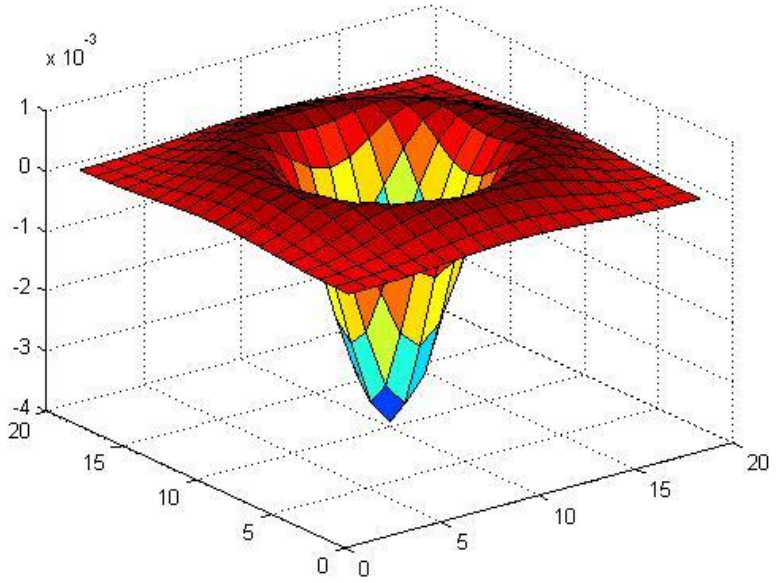
- Instead of computing *f* for larger and larger windows, we can implement using a fixed window size with a Gaussian pyramid

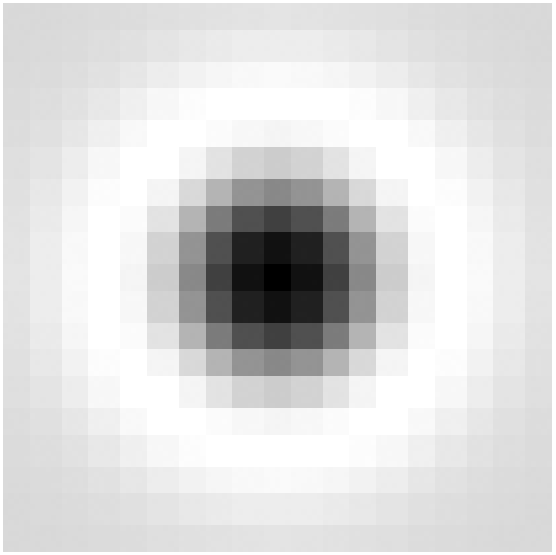


(sometimes need to create in-between levels, e.g. a ¾-size image)

Another Common Definition of *f*

- The *Laplacian of Gaussian* (*LoG*)





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

(very similar to a Difference of Gaussians (DoG) – i.e. a Gaussian minus a slightly smaller Gaussian)

Another Common Definition of *f*

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

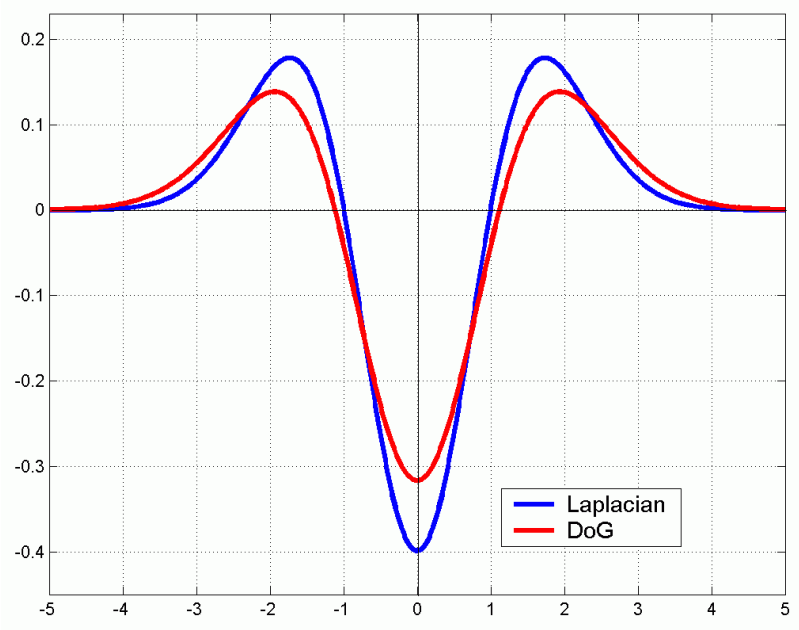
Kernels:

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

(Laplacian of Gaussian)

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

(Difference of Gaussians)



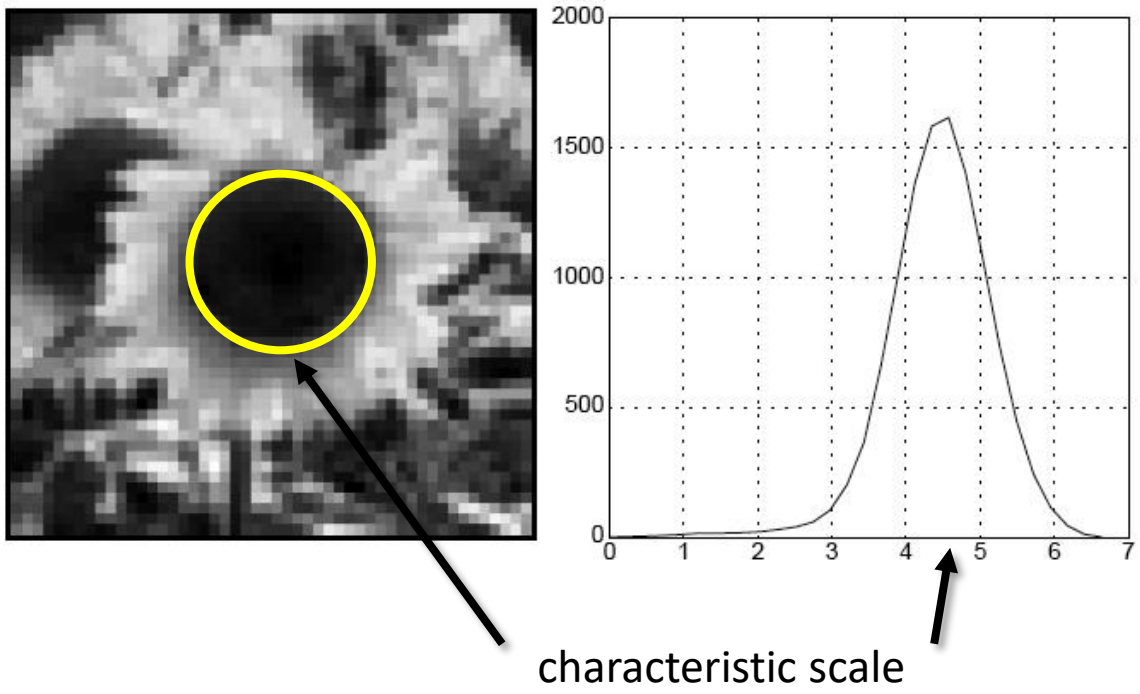
where Gaussian

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

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

Characteristic Scale

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



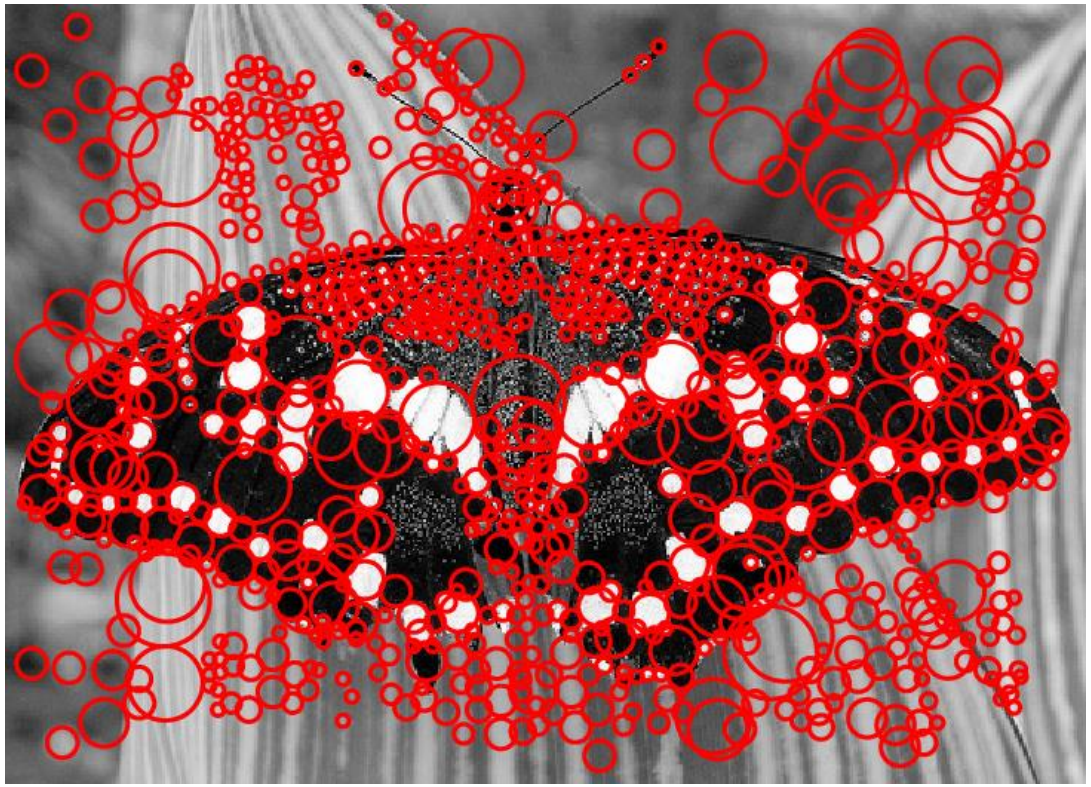
T. Lindeberg (1998). "Feature detection with automatic scale selection." *International Journal of Computer Vision* 30 (2): pp 77--116.

scale-space blob detector: example

scale-space blob detector: example

sigma = 11.9912

scale-space blob detector: example

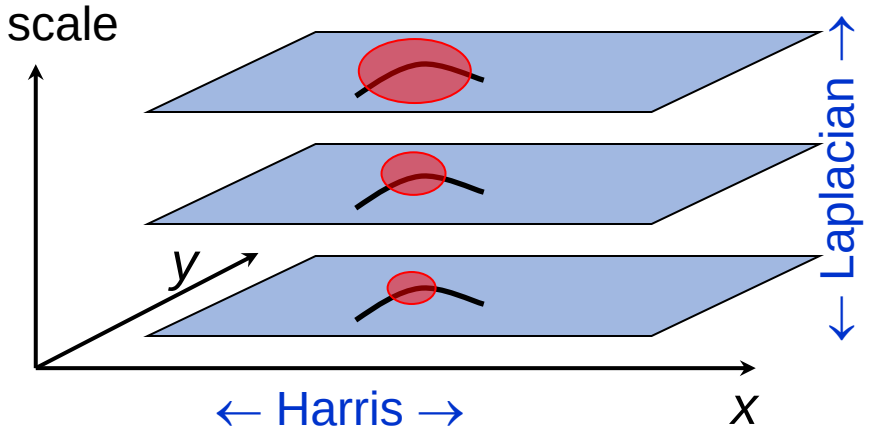


Scale Invariant Detectors

- Harris-Laplacian[1]

  Find local maximum of:

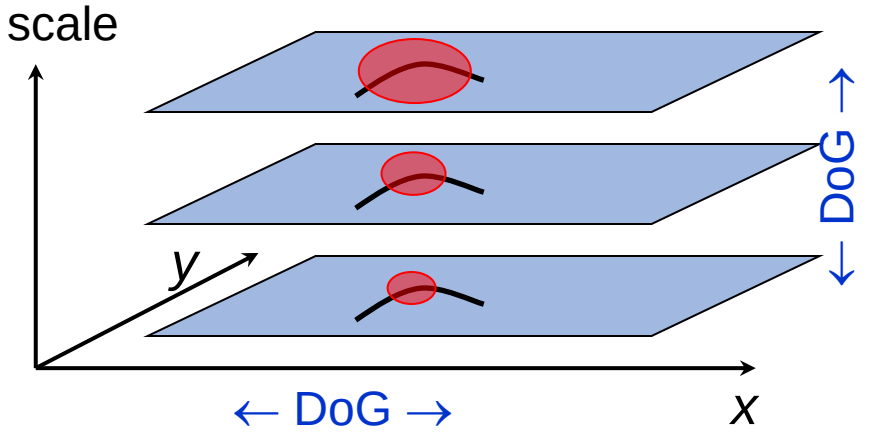
  - Harris corner detector in space (image coordinates)
  - Laplacian in scale



- SIFT (Lowe)[2]

  Find local maximum of:

  - Difference of Gaussians in space and scale



[1] K.Mikolajczyk, C.Schmid. "Indexing Based on Scale Invariant Interest Points". ICCV 2001

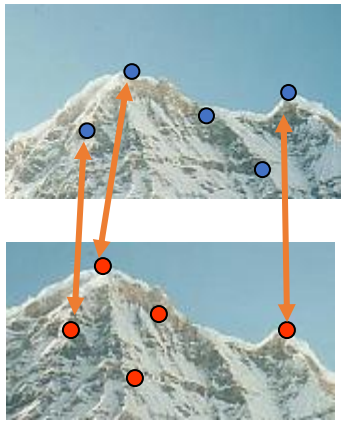
[2] D.Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". Accepted to IJCV 2004

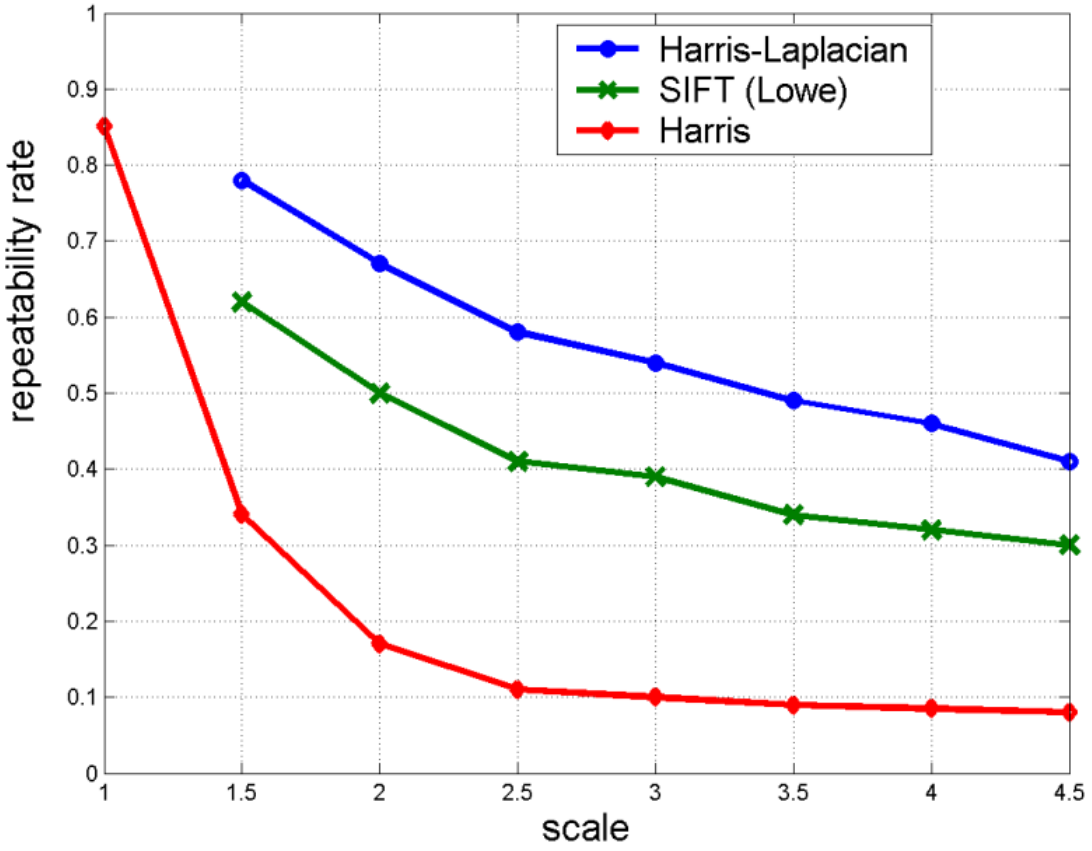
Scale Invariant Detectors

- Experimental evaluation of detectors w.r.t. scale change

Repeatability rate:

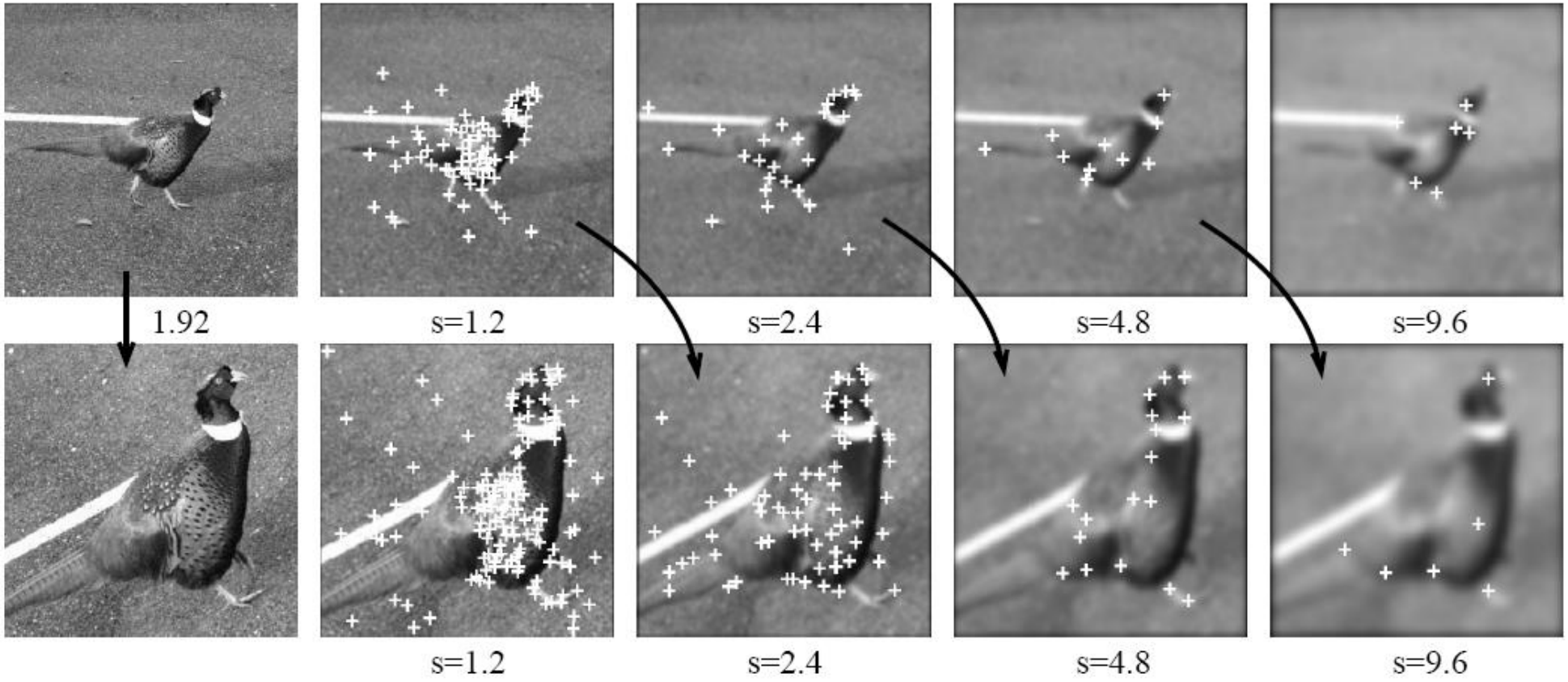
$$\frac{\#\text{ correspondences}}{\#\text{ possible correspondences}}$$





K.Mikolajczyk, C.Schmid. “Indexing Based on Scale Invariant Interest Points”. ICCV 2001

Examples of Harris-Laplacian Features



Questions?

FAST Corners

Machine learning for high-speed corner detection

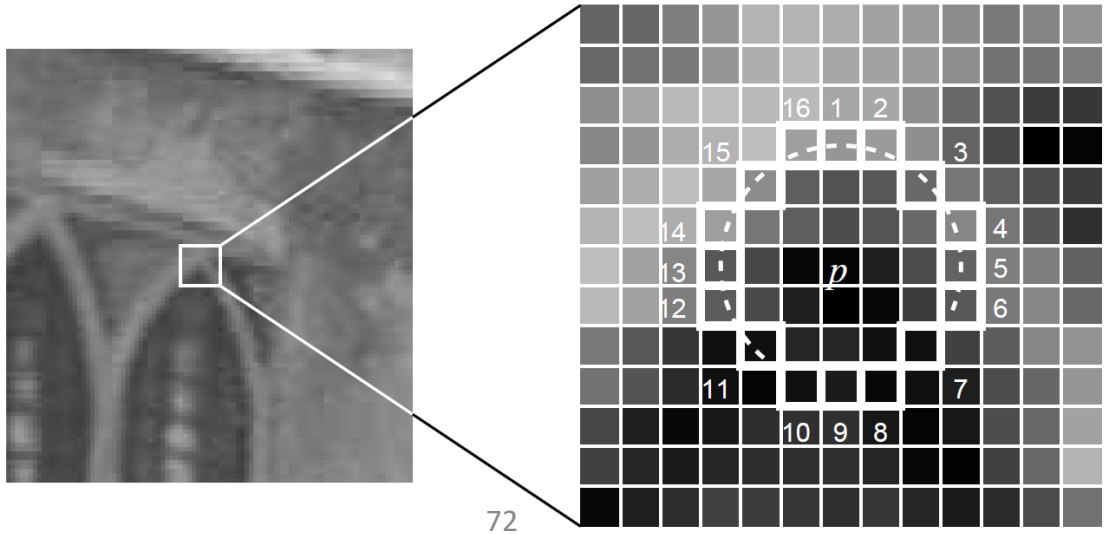
Edward Rosten and Tom Drummond

Department of Engineering, Cambridge University, UK
{er258, twd20}@cam.ac.uk

2006 ECCV

Detecting Corners by Decision Trees

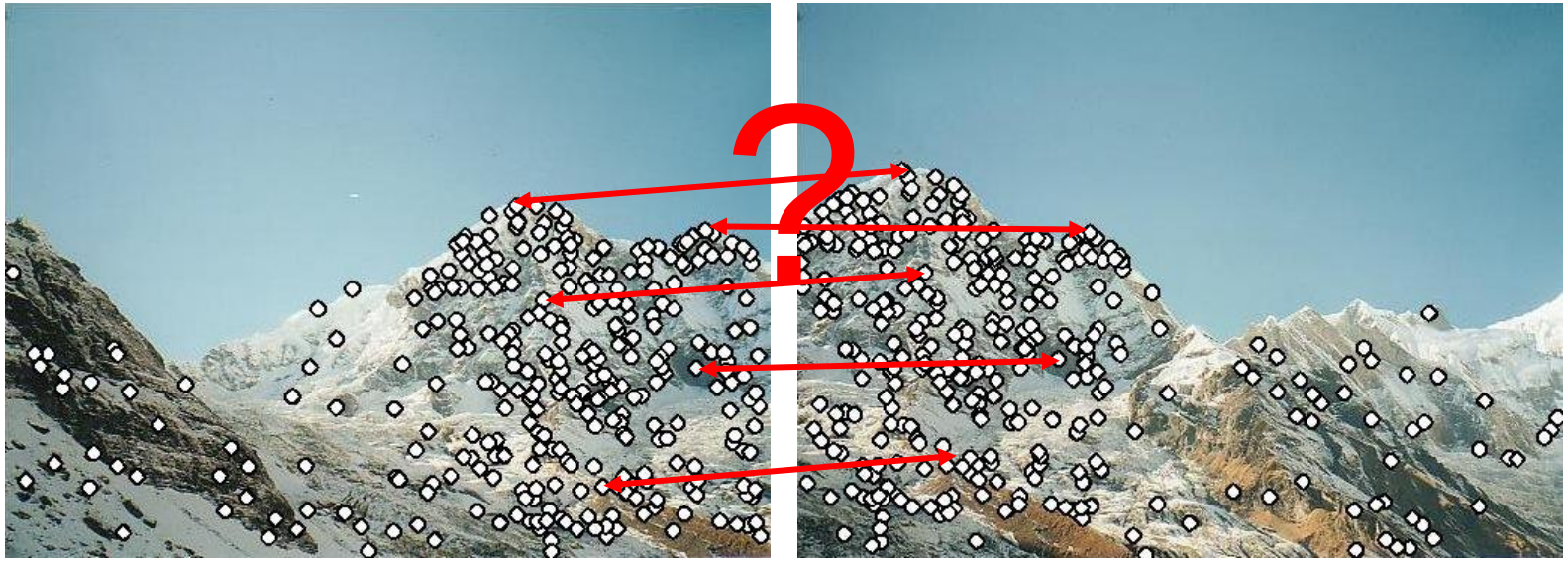
- A pixel p is a corner, if:
  - There are n continuous pixels in a circle that are brighter/darker than I_p
  - Brighter and darker is measured by another threshold t
- If $n = 12$, we can check I_1, I_5, I_9, I_{13} (four compass directions first) to by-pass many non-corners
- FAST detector use a decision tree to make the whole process efficient



Questions

Feature Descriptors

- We know how to detect good points
- Next question: **How to match them?**



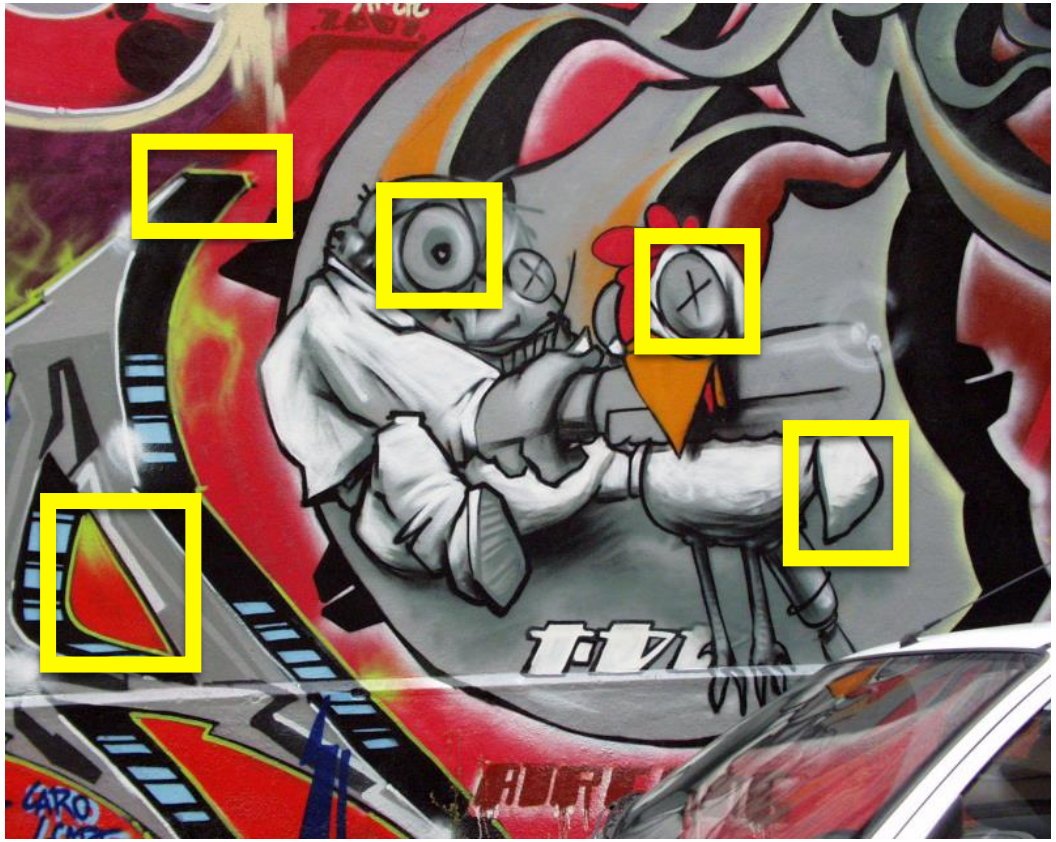
Answer: Come up with a *descriptor* for each point,
find similar descriptors between the two images

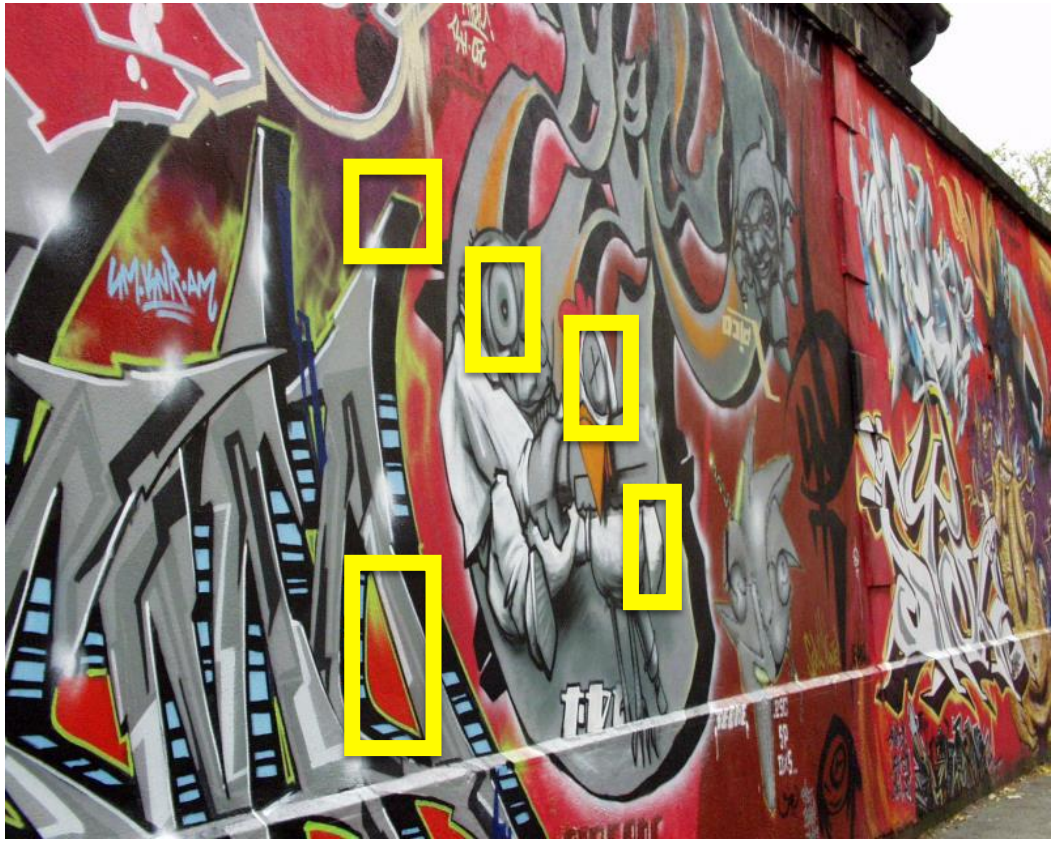
Lots of possibilities (this is a popular research area)

Simple option: match square windows around the point

A popular choice: SIFT David Lowe, UBC http://www.cs.ubc.ca/~lowe/keypoints/

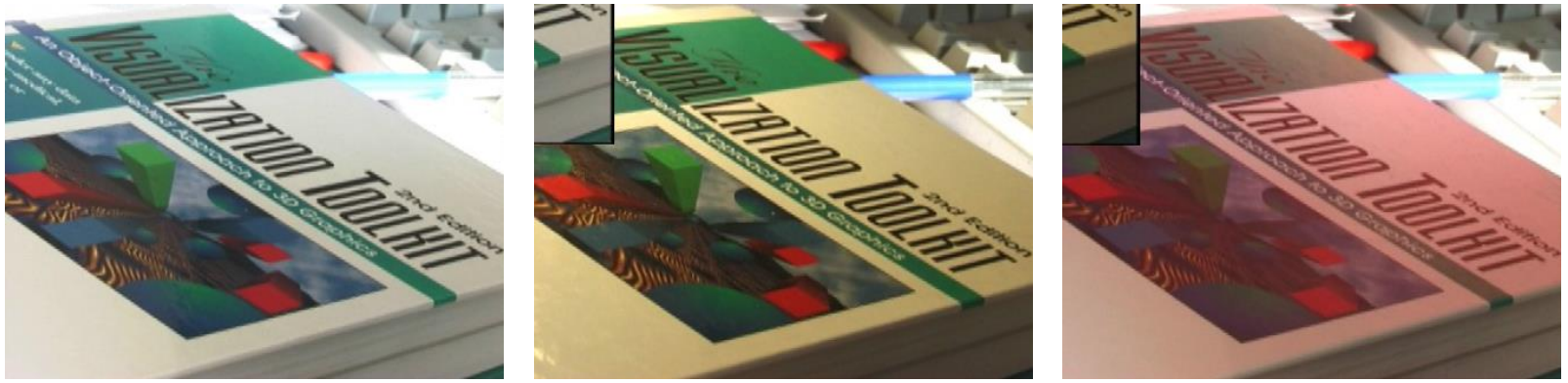
how do we describe a local image patch



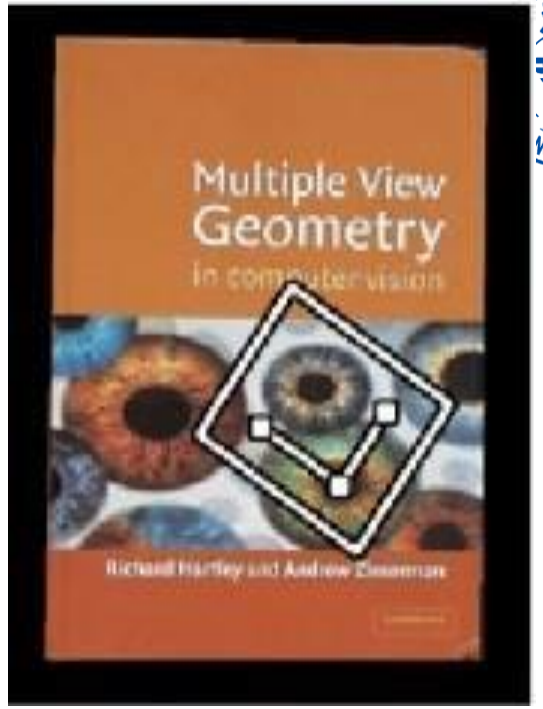


Invariance vs. discriminability

- Invariance:
  - Descriptor shouldn't change even if image is transformed
- Discriminability:
  - Descriptor should be highly unique for each point



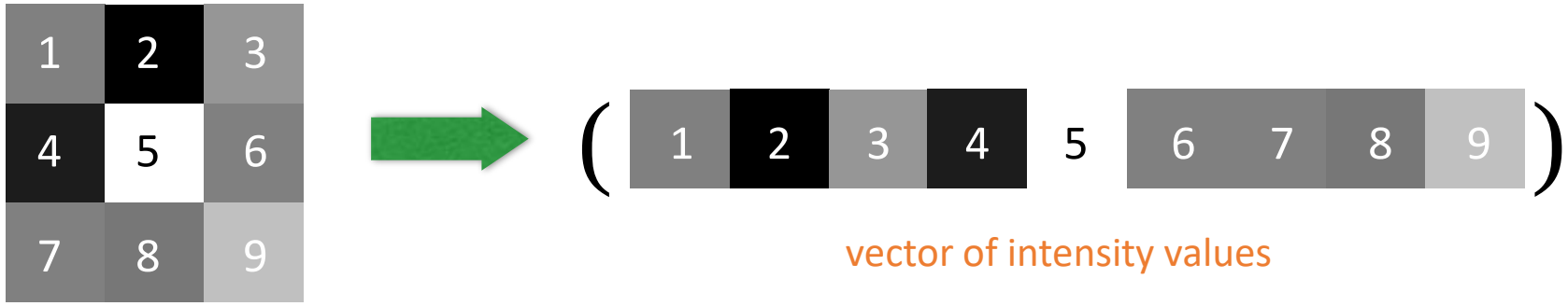
photometric transformations



geometric transformations

Image Patch

- Just use the pixel values of the patch



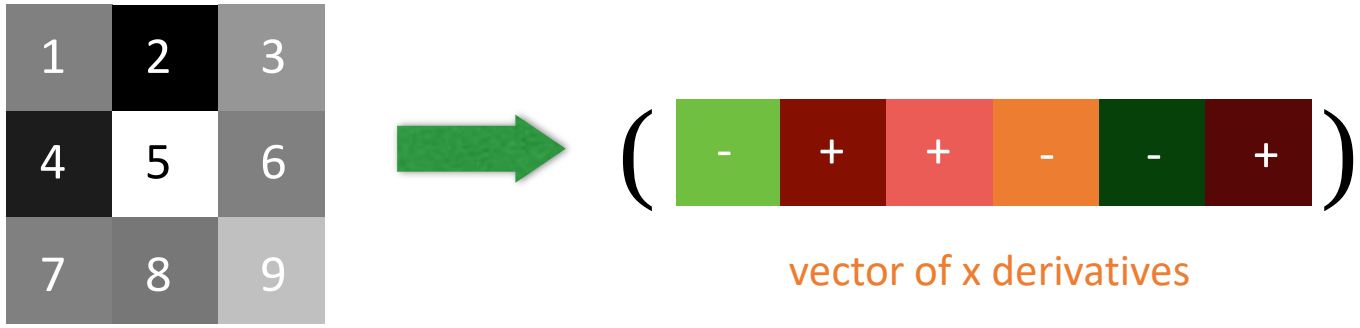
Perfectly fine if geometry and appearance is unchanged (a.k.a. template matching)

What are the problems?

How can you be less sensitive to absolute intensity values?

Image Gradients

- Use pixel differences



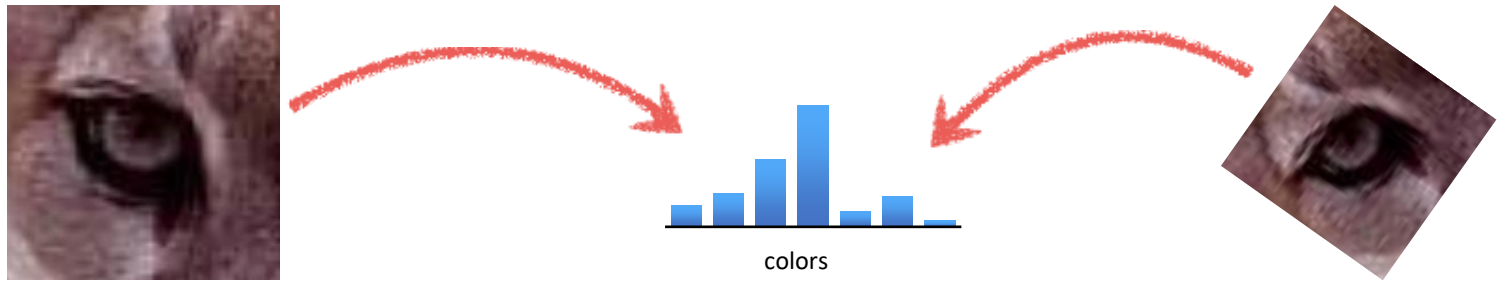
Feature is invariant to absolute intensity values

What are the problems?

How can you be less sensitive to deformations?

Color Histogram

- Count the colors in the image using a histogram

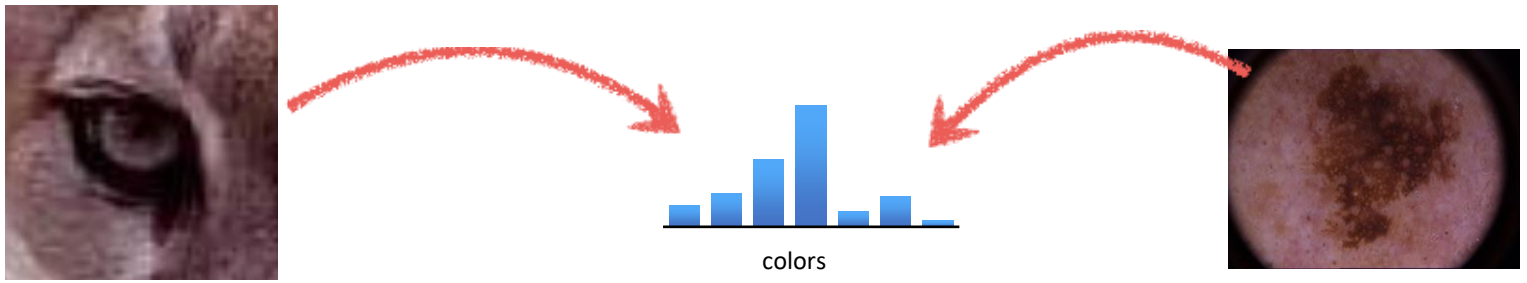


Invariant to changes in scale and rotation

What are the problems?

Color Histogram

- Count the colors in the image using a histogram

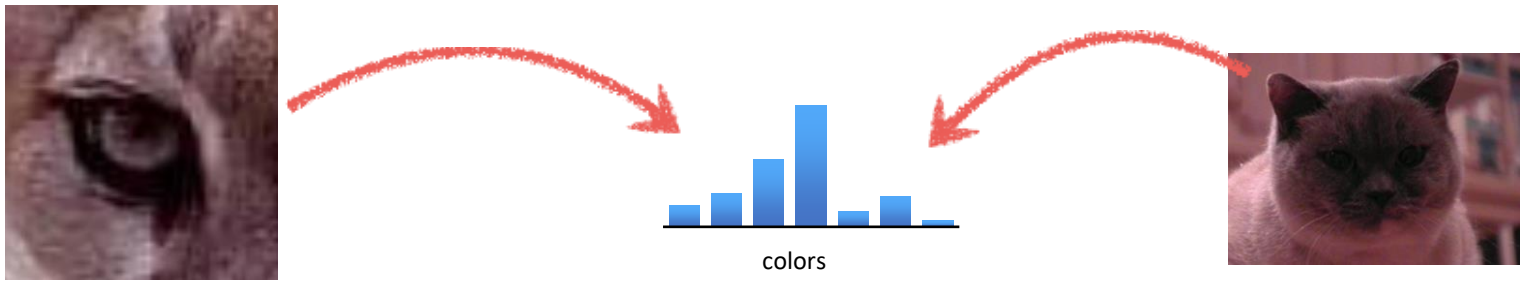


Invariant to changes in scale and rotation

What are the problems?

Color Histogram

- Count the colors in the image using a histogram



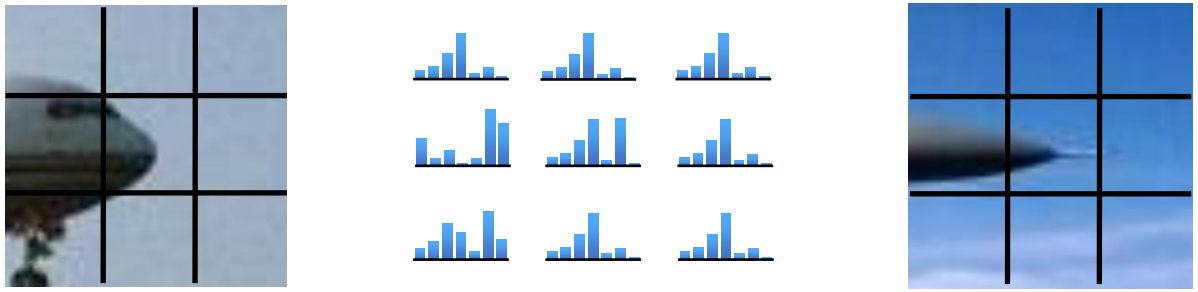
Invariant to changes in scale and rotation

What are the problems?

How can you be more sensitive to spatial layout?

Spatial Histograms

- Compute histograms over spatial ‘cells’



Retains rough spatial layout

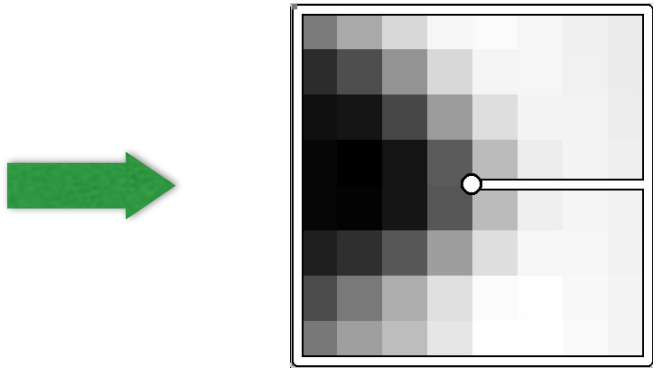
Some invariance to deformations

What are the problems?

How can you be completely invariant to rotation?

Orientation Normalization

- Use the dominant image gradient direction to normalize the orientation of the patch



save the orientation angle θ along with (x, y, s)

What are the problems?

Questions?

Invariance

- Most feature descriptors are designed to be invariant to
  - Translation, 2D rotation, scale

- They can usually also handle
  - Limited 3D rotations (SIFT works up to about 60 degrees)
  - Limited affine transformations (some are fully affine invariant)
  - Limited illumination/contrast changes

Rotation Invariance for Feature Descriptors

- Find dominant orientation of the image patch
  - This could be computed in several ways:
    - $\mathbf{x}_{max}$, the eigenvector of **M** corresponding to λ_{max}
    - Smoothed gradient direction at the center point
  - Rotate the patch according to this angle

$$M = \begin{pmatrix} A & B \\ B & C \end{pmatrix}$$

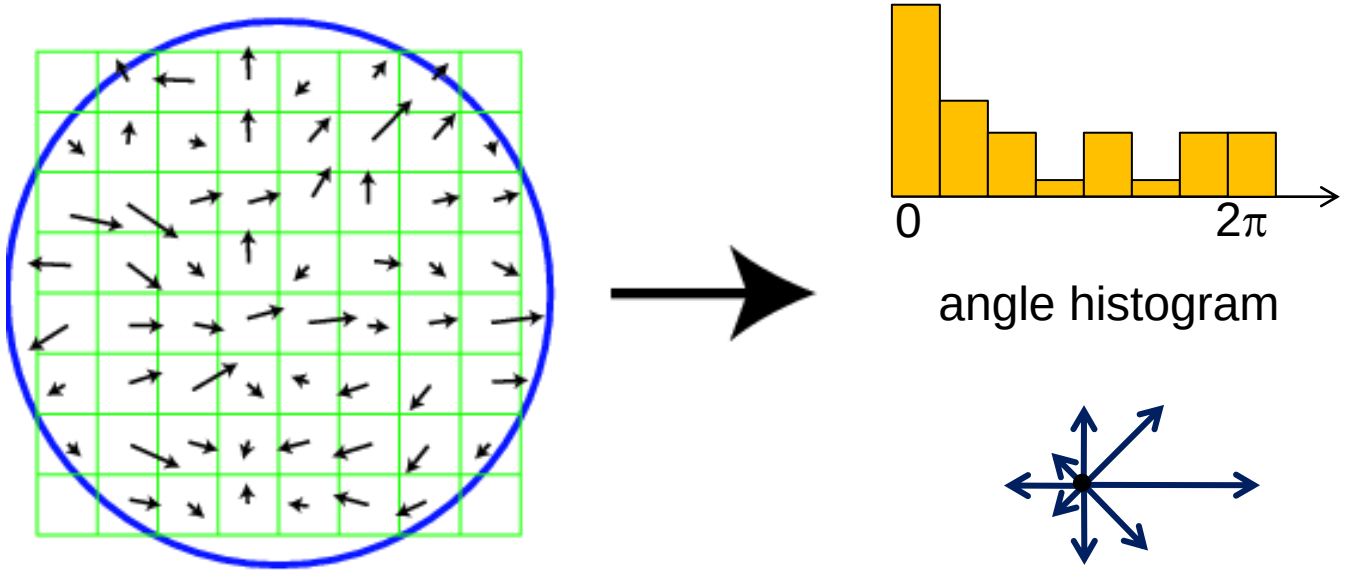
$$A = \sum_{x,y} w(x,y) I_x^2 , \quad C = \sum_{x,y} w(x,y) I_y^2$$

$$B = \sum_{x,y} w(x,y) I_x \, I_y ,$$

Scale Invariant Feature Transform

Basic idea:

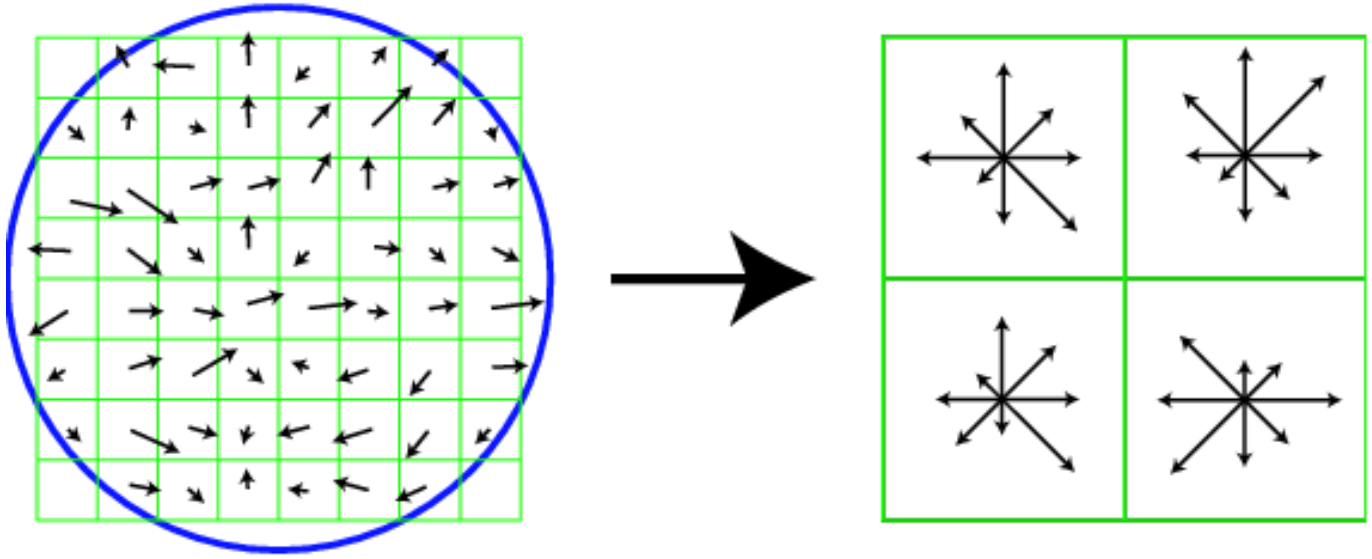
- Take 16x16 square window around detected feature
- Compute edge orientation (angle of the gradient – 90^o) for each pixel
- Throw out weak edges (threshold gradient magnitude)
- Create histogram of surviving edge orientations



SIFT descriptor

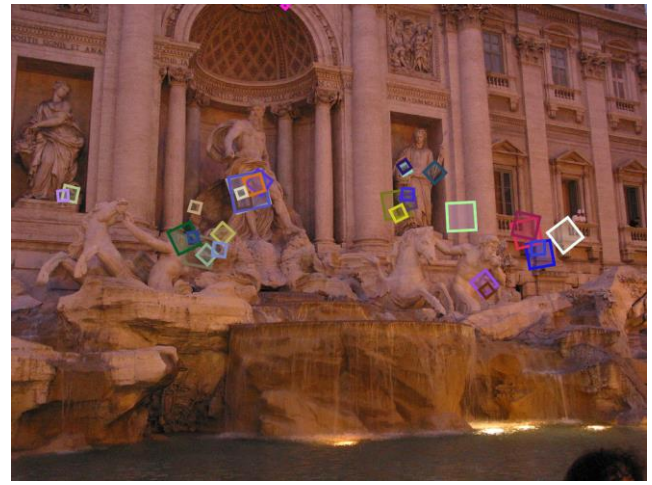
Full version

- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
- Compute an orientation histogram for each cell
- 16 cells * 8 orientations = 128 dimensional descriptor



Properties of SIFT

- Can handle changes in viewpoint
  - Up to about 60 degree out of plane rotation
- Can handle significant changes in illumination
  - Sometimes even day vs. night (right)
- Fast and efficient—can run in real time
- Lots of code available
  - http://people.csail.mit.edu/albert/ladypack/wiki/index.php/Known_implementations_of_SIFT



SURF: Speeded Up Robust Features

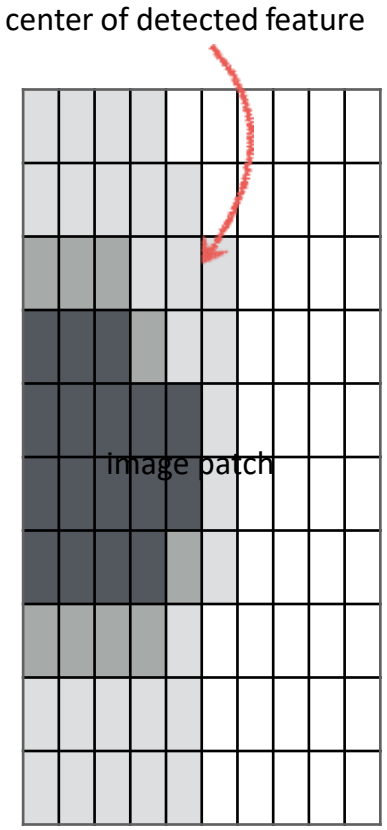
Herbert Bay[1], Tinne Tuytelaars[2], and Luc Van Gool[12]

[1] ETH Zurich
{bay, vangool}@vision.ee.ethz.ch
[2] Katholieke Universiteit Leuven
{Tinne.Tuytelaars, Luc.Vangool}@esat.kuleuven.be

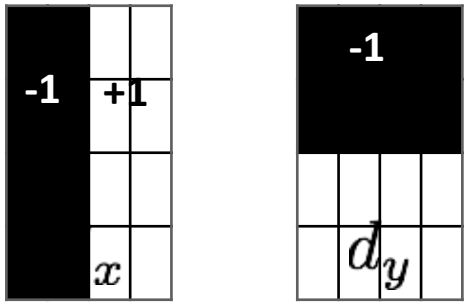
ECCV 2006

SURF('Speeded' Up Robust Features)

Compute Haar wavelet response at each pixel in patch



Haar wavelets filters



(Gaussian weighted from center)

How would do you compute the filter response?

Filtering using a sliding window can be slow
Haar wavelets are just sums over blocks
Use integral images for efficiency (6 operations)

Integral Image

original image

$I(x, y)$

1	5	2
2	4	1
2	1	1

$A(x, y)$

1	6	8
3	12	15
5	15	19

integral image

$$A(x, y) = \sum_{x' \leq x, y' \leq y} I(x', y')$$

Integral Image

original image

$I(x, y)$

1	5	2
2	4	1
2	1	1

$A(x, y)$

1	6	8
3	12	15
5	15	19

integral image

$$A(x, y) = \sum_{x' \le x, y' \le y} I(x', y')$$

Can find the **sum** of any block using **3** operations

$$A(x_1, y_1, x_2, y_2) = A(x_2, y_2) - A(x_1, y_2) - A(x_2, y_1) + A(x_1, y_1)$$

What is the sum of the bottom right 2x2 square?

$$A(x_1, y_1, x_2, y_2) = A(x_2, y_2) - A(x_1, y_2) - A(x_2, y_1) + A(x_1, y_1)$$

$I(x, y)$

1	5	2
2	4	1
2	1	1

image

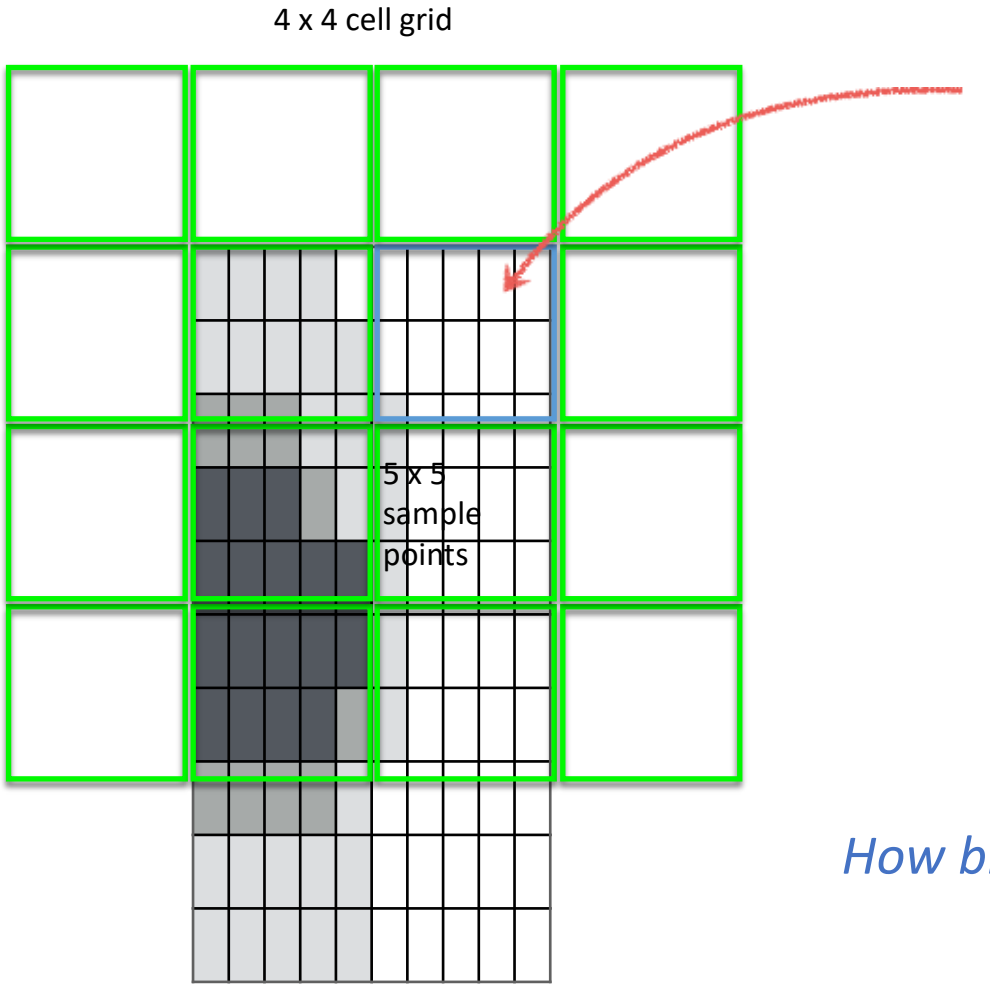
$A(x, y)$

1	6	**8**
3	12	15
5	15	**19**

integral image

$$\begin{aligned} A(1,1,3,3) &= A(3,3) - A(1,3) - A(3,1) + A(1,1) \\ &= 19 - 8 - 5 + 1 \\ &= 7 \end{aligned}$$

SURF('Speeded' Up Robust Features)

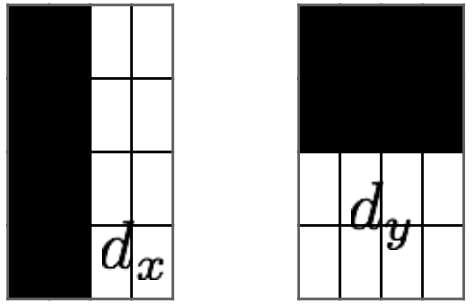


Each cell is represented by 4 values:

$$\left[\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|\right]$$

Haar wavelets filters

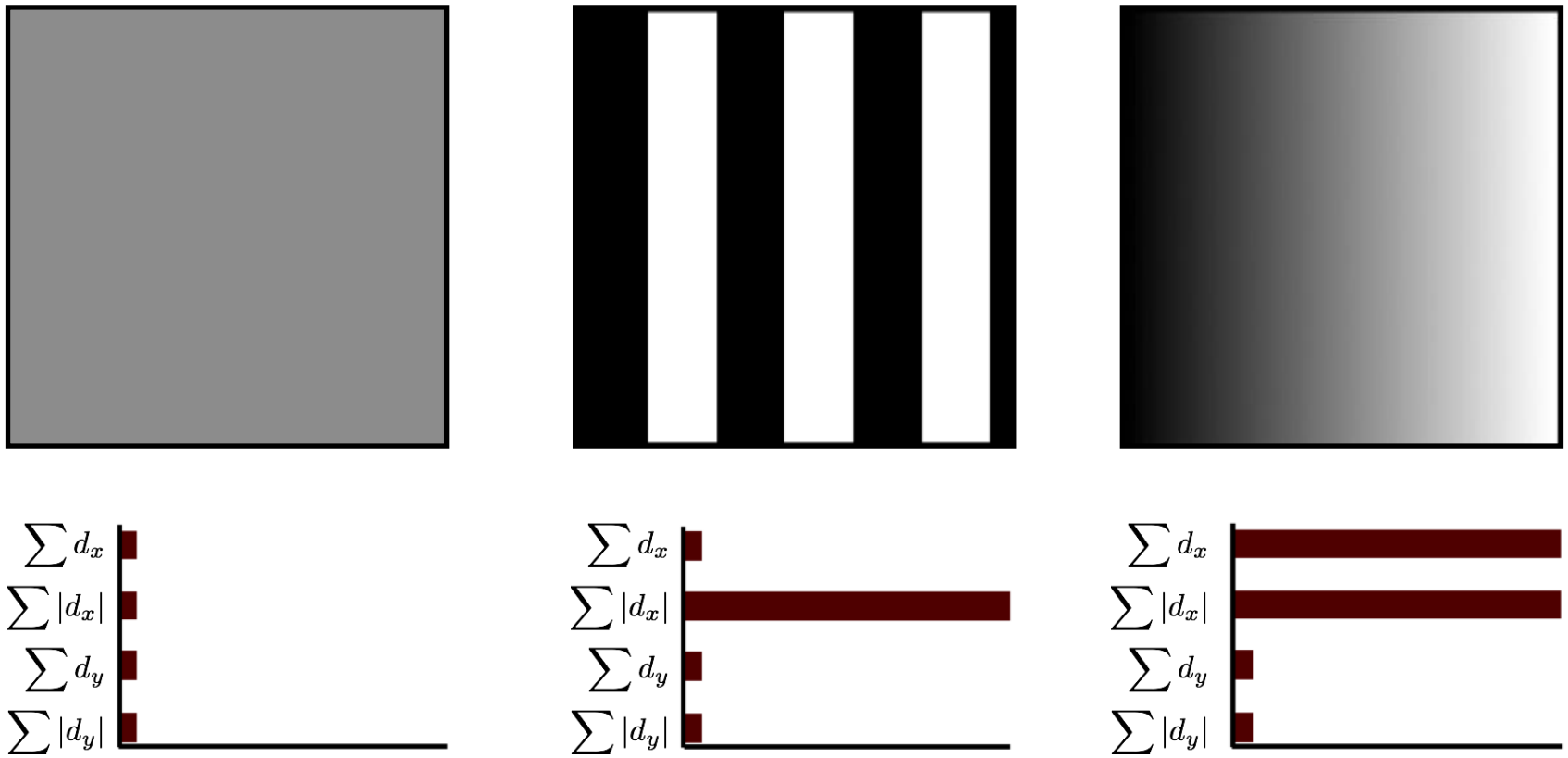
(Gaussian weighted from center)



How big is the SURF descriptor?

64 dimensions

examples


$\sum d_x$
$\sum |d_x|$
$\sum d_y$
$\sum |d_y|$
$\sum d_x$
$\sum |d_x|$
$\sum d_y$
$\sum |d_y|$
$\sum d_x$
$\sum |d_x|$
$\sum d_y$
$\sum |d_y|$

BRIEF: Binary Robust Independent Elementary Features*

Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua

CVLab, EPFL, Lausanne, Switzerland
e-mail: firstname.lastname@epfl.ch

ECCV 2010

BRIEF(Binary Robust Independent Elementary Features)

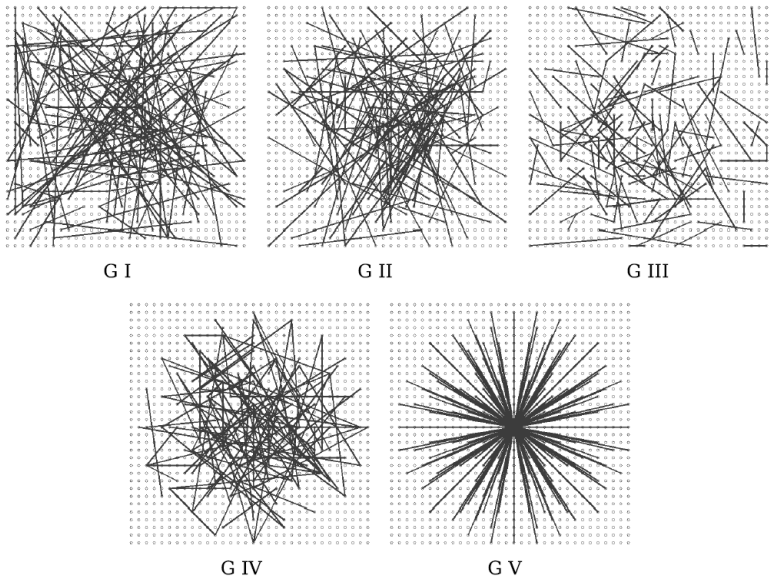
- Smooth the image by Gaussian (e.g. $\sigma = 2$)
- Sample two pixels $\boldsymbol{x}, \boldsymbol{y}$ within the local region and compare their relative values

$$\tau(\boldsymbol{x}; \boldsymbol{y}) = \begin{cases} 1 & if\ I(\boldsymbol{x}) < I(\boldsymbol{y}) \\ 0 & otherwise \end{cases}$$

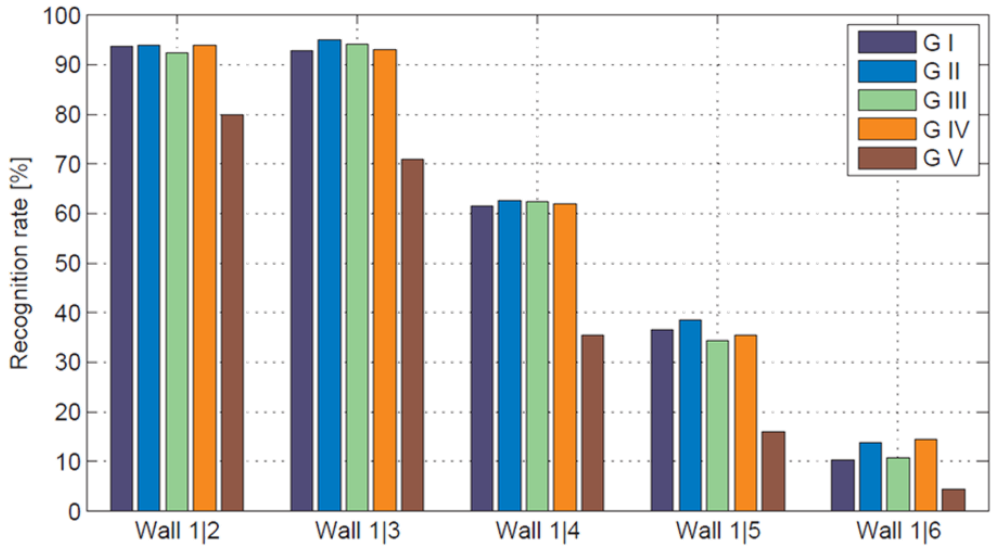
- Generate 256/512 such pairs to generate a 256/512-bit descriptor
- Use the Hamming distance to compare two descriptors

BRIEF(Binary Robust Independent Elementary Features)

- Test different ways to sample pixel pairs $\boldsymbol{x}, \boldsymbol{y}$
- Random sampling works best



different sampling strategies



performance of different sampling

ORB: an efficient alternative to SIFT or SURF

Ethan Rublee Vincent Rabaud Kurt Konolige Gary Bradski

Willow Garage, Menlo Park, California

{erublee}{vrabaud}{konolige}{bradski}@willowgarage.com

ICCV 2011

ORB(Oriented FAST and Rotated BRIEF)

- A combination of FAST and BRIEF
- Compute orientation of FAST feature (by a fast method)
  - Denote the orientation by θ
- When computing the BRIEF descriptor, rotate the positions of x, y by the angle θ

$$S = \begin{pmatrix} x_1 & x_2 & \cdots \\ y_1 & y_2 & \cdots \end{pmatrix} \rightarrow S_\theta = R_\theta \begin{pmatrix} x_1 & x_2 & \cdots \\ y_1 & y_2 & \cdots \end{pmatrix}$$

Questions?

Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope*

AUDE OLIVA
Harvard Medical School and the Brigham and Women's Hospital, 221 Longwood Ave., Boston, MA 02115
oliva@search.bwh.harvard.edu

ANTONIO TORRALBA
Department of Brain and Cognitive Sciences, MIT, 45 Carleton Street, Cambridge, MA 02139
torralba@ai.mit.edu

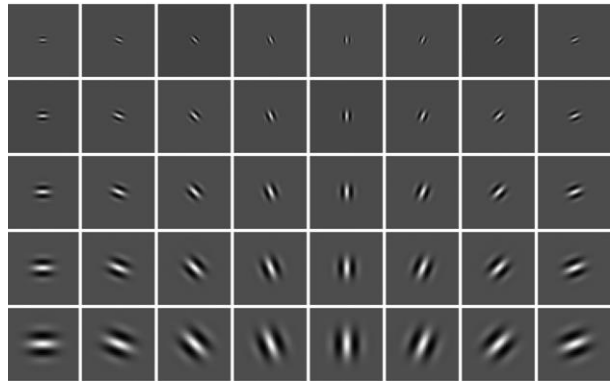
IJCV 2001

Scene Recognition and Holistic Image Feature

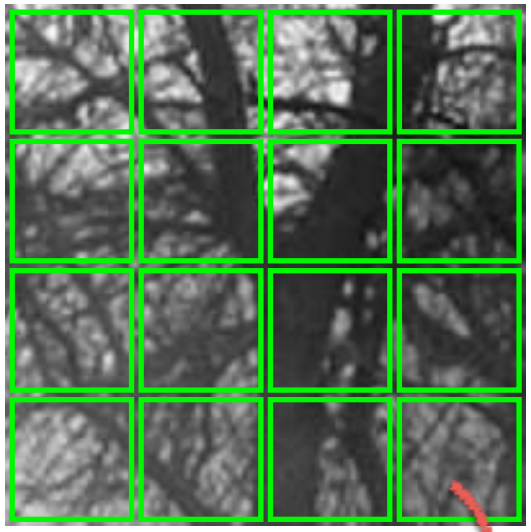
- Some psychophysical studies suggest humans recognize a scene category very quickly (e.g. ~200 ms)
  - Object details are often ignored
  - Some holistic image features might be used by humans
- GIST is such a holistic image feature
  - Measure the overall content of an image
  - E.g. used in image search in the past

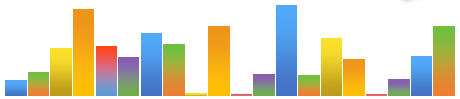
GIST

- Compute filter responses (filter bank of Gabor filters)
- Divide image patch into 4×4 cells
- Compute filter response averages for each cell
- Size of descriptor is $4 \times 4 \times N$, where N is the size of the filter bank



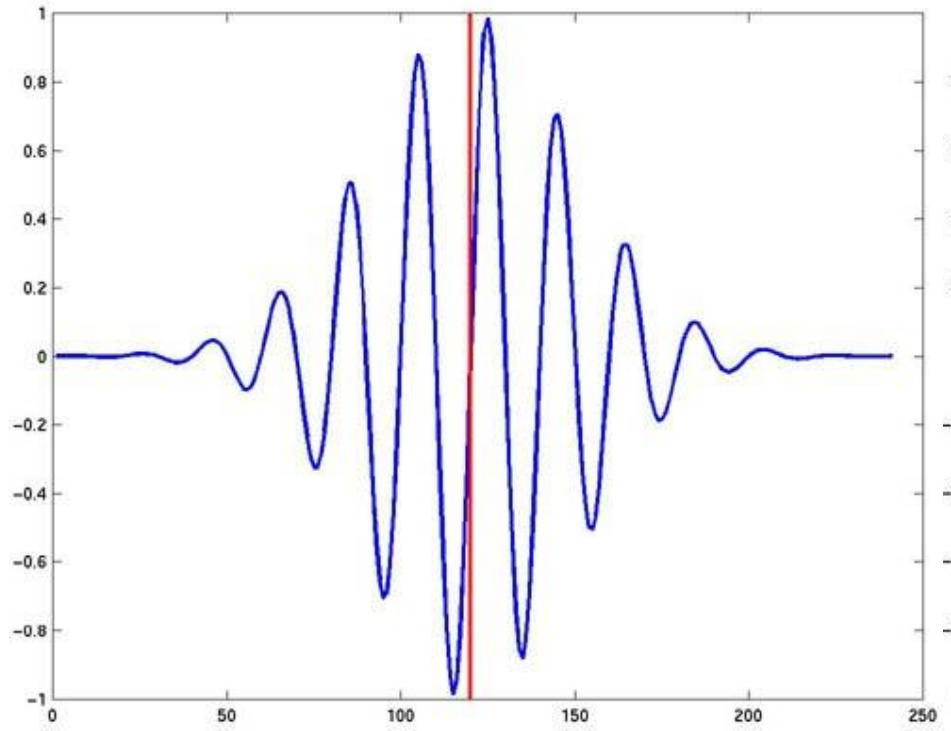
4 x 4 cell

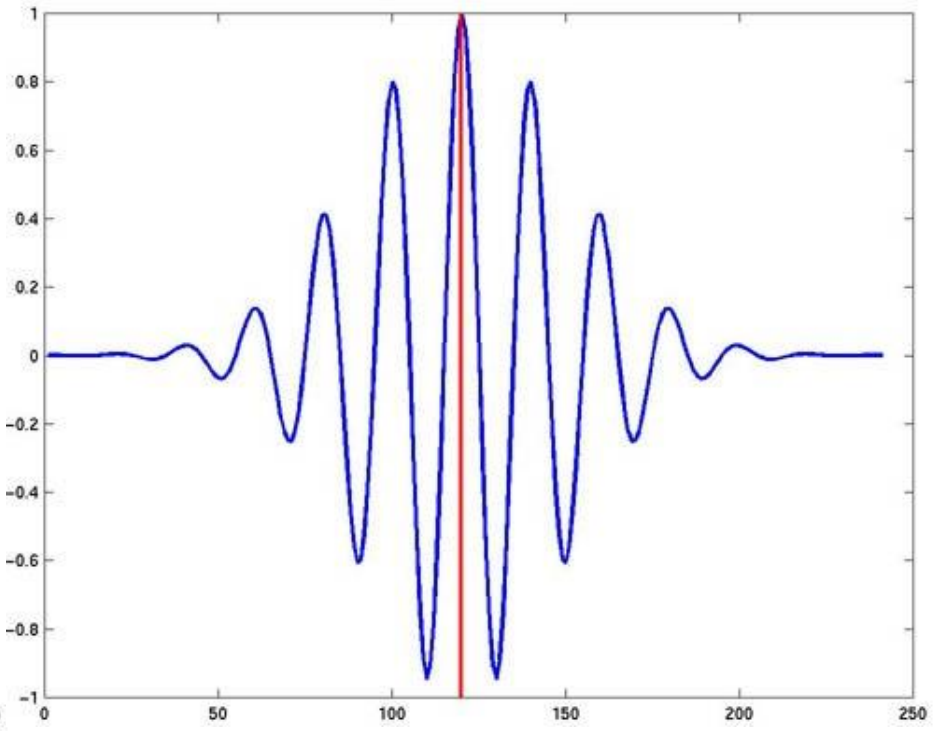




averaged filter responses

Gabor Filters (1D examples)

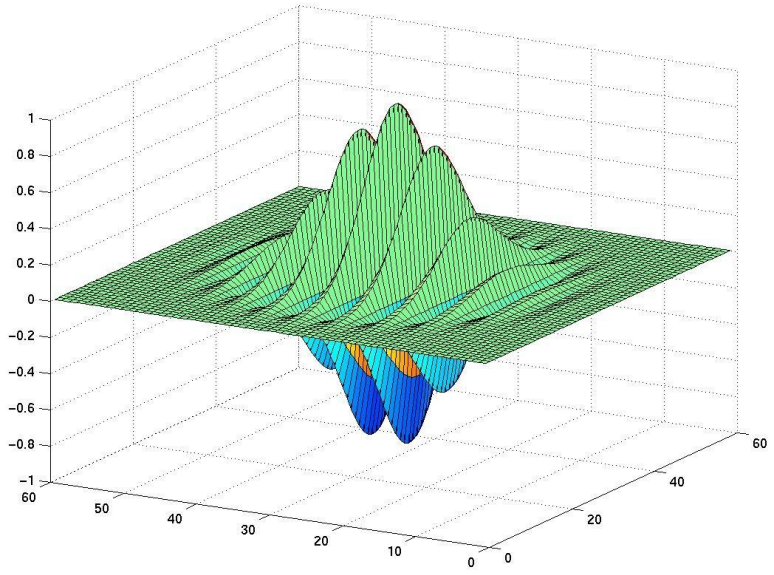


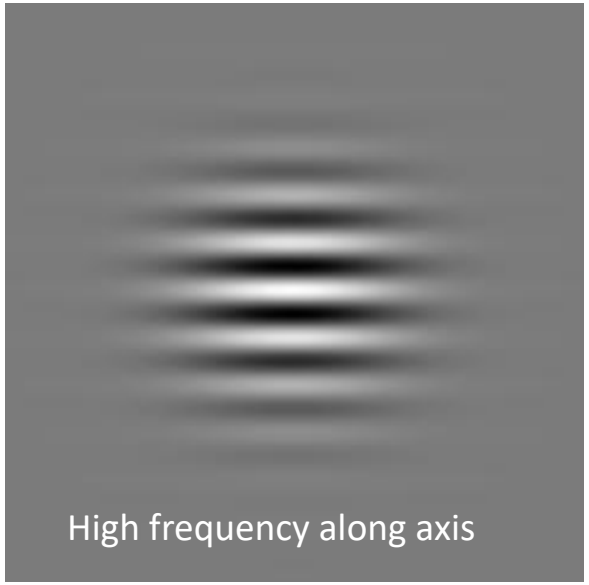


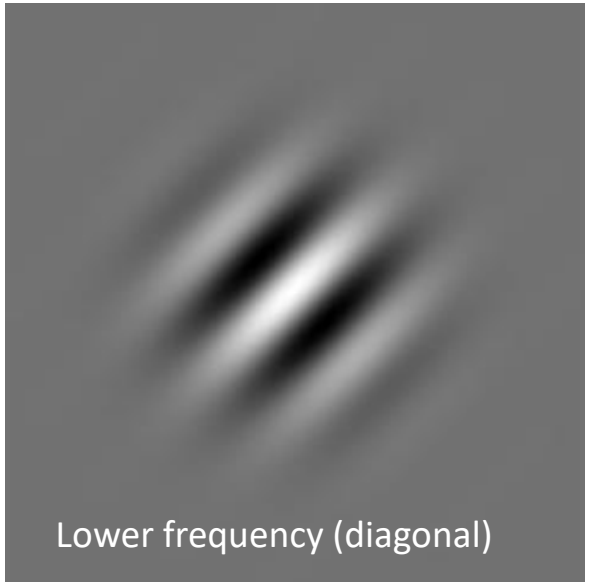
$$e^{-\frac{x^2}{2\sigma^2}}\sin(2\pi\omega x) \qquad e^{-\frac{x^2}{2\sigma^2}}\cos(2\pi\omega x)$$

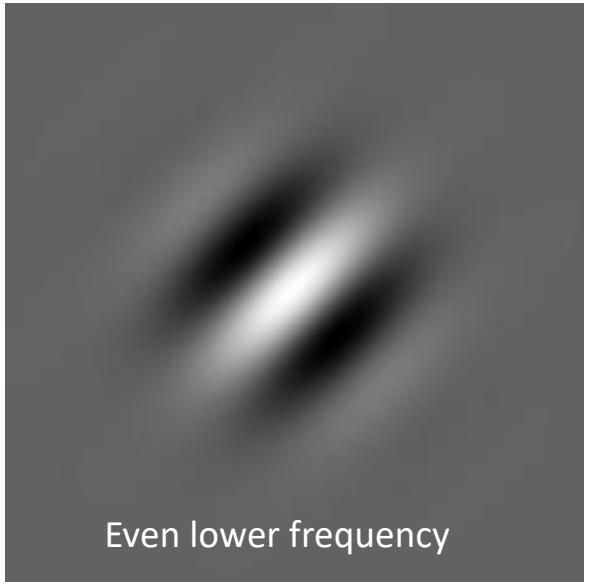
2D Gabor Filters

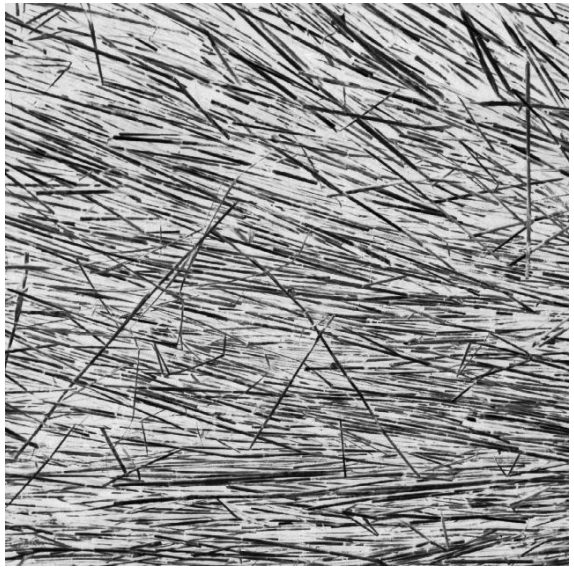
$$e^{-\frac{x^2+y^2}{2\sigma^2}} \cos\left(2\pi\left(k_x x + k_y y\right)\right)$$

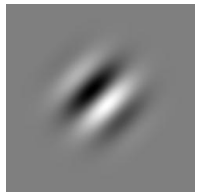


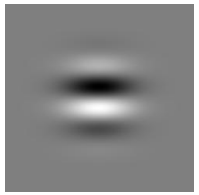

High frequency along axis

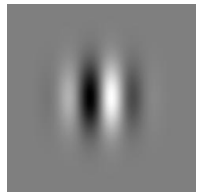

Lower frequency (diagonal)

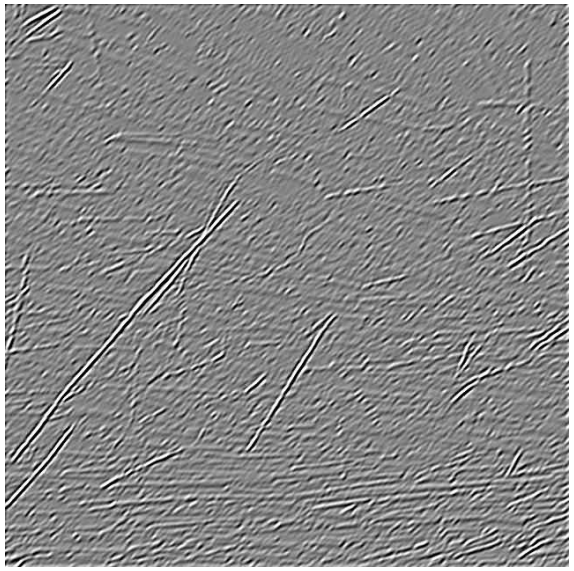

Even lower frequency

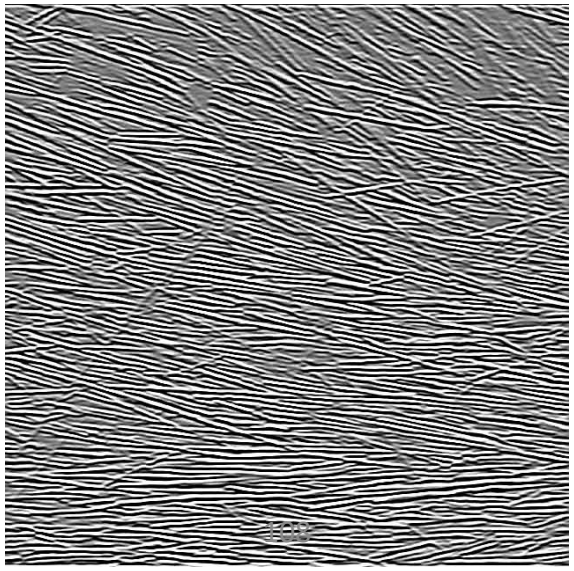


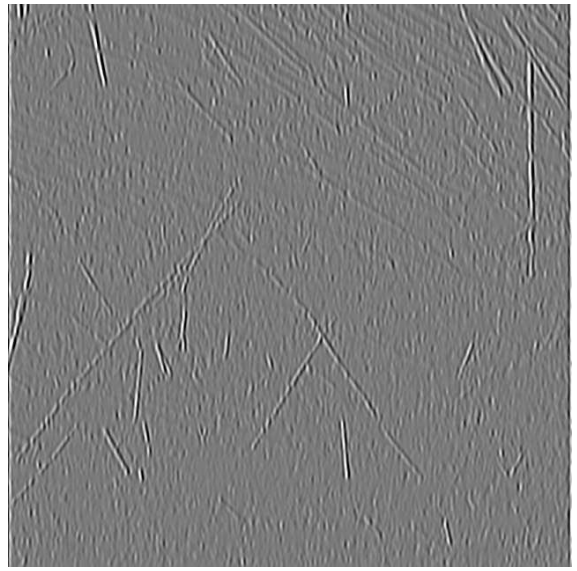


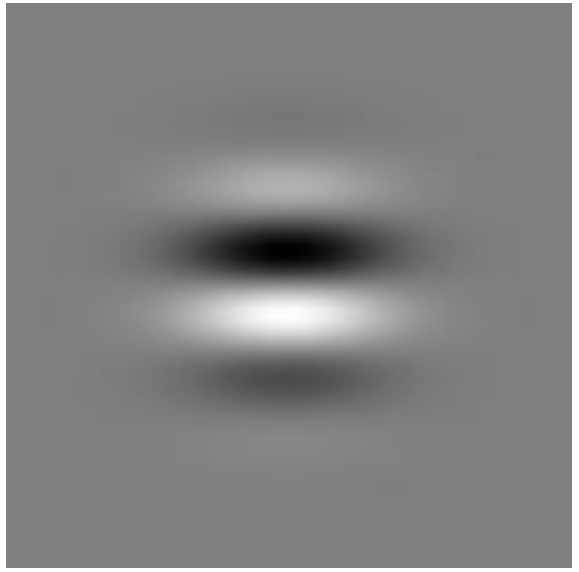




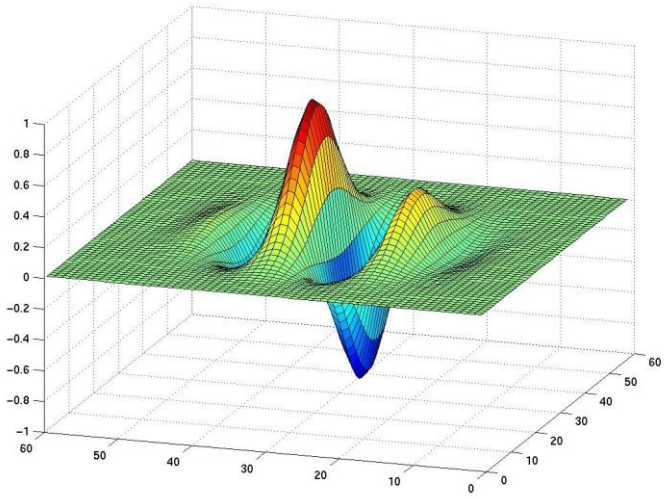




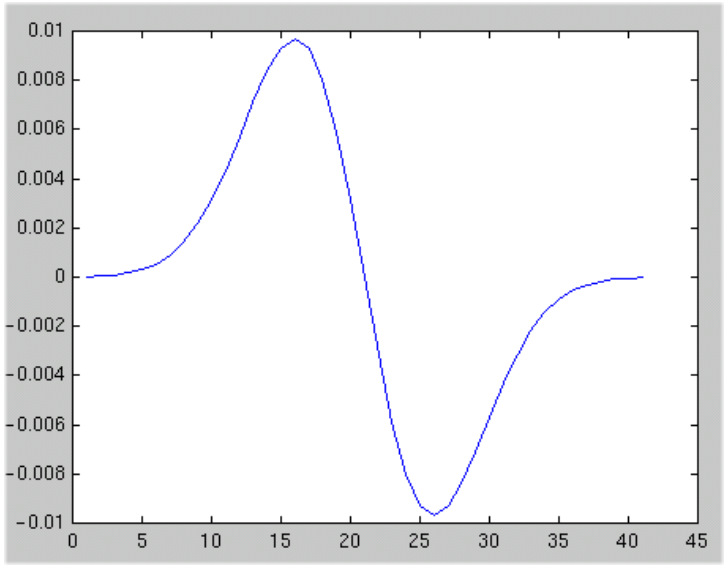




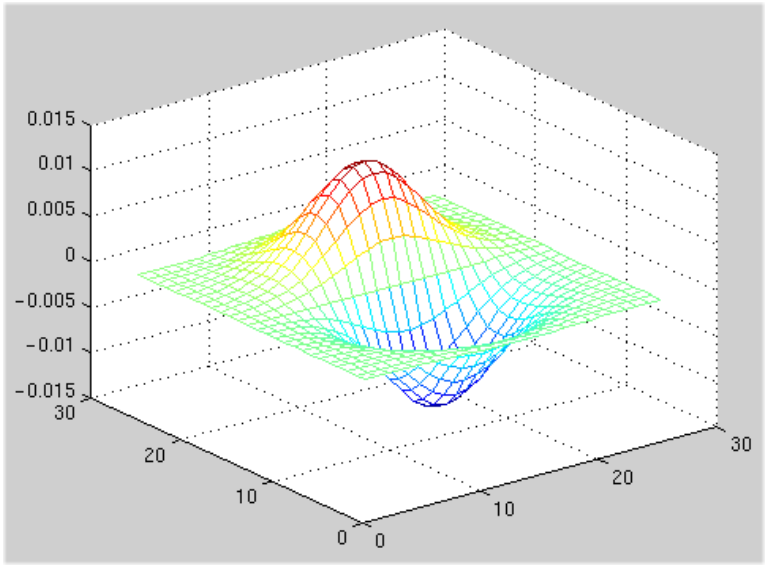
Odd Gabor filter

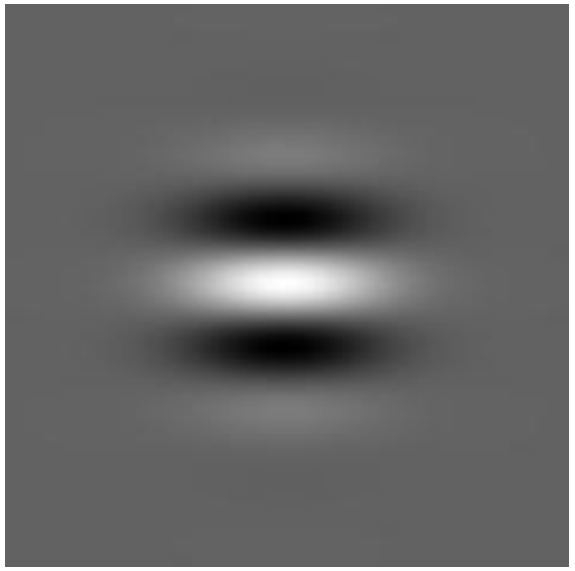


… looks a lot like…

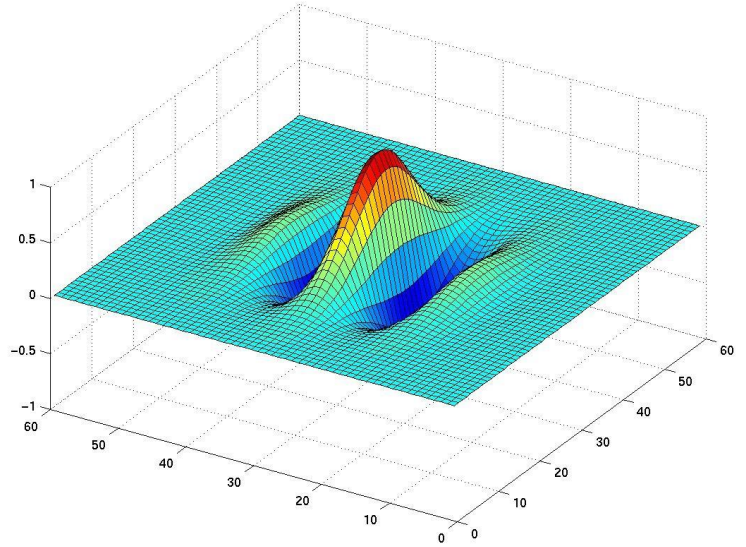


Gaussian Derivative

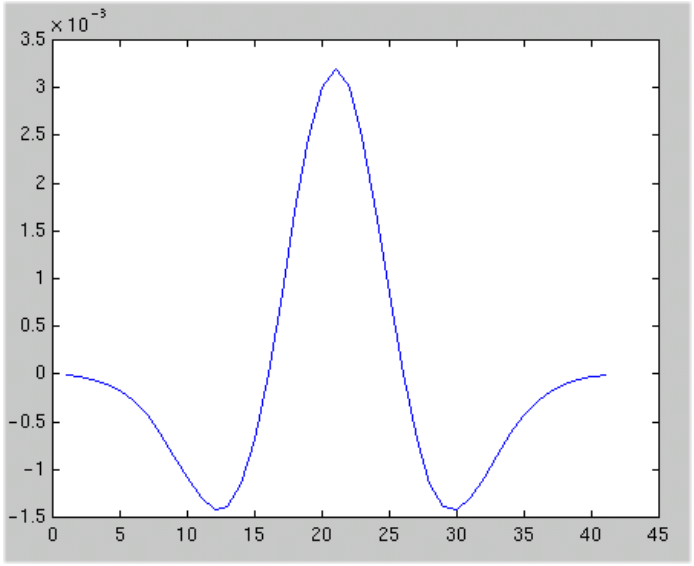




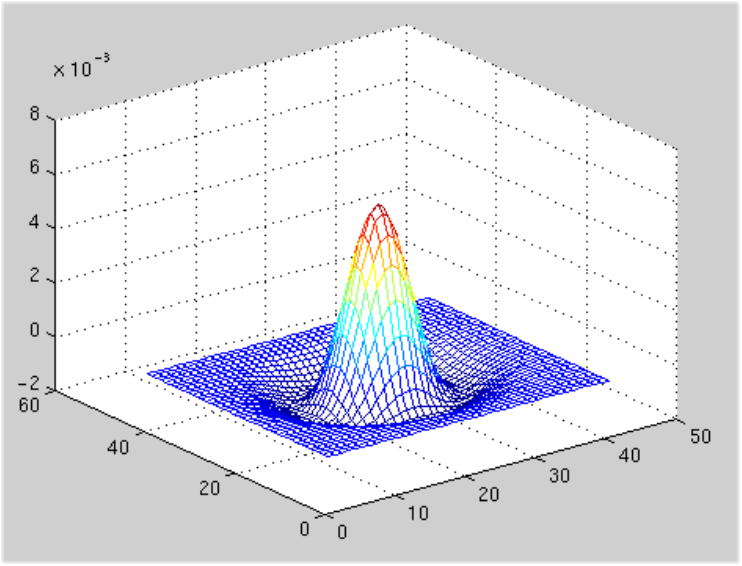
Even Gabor filter



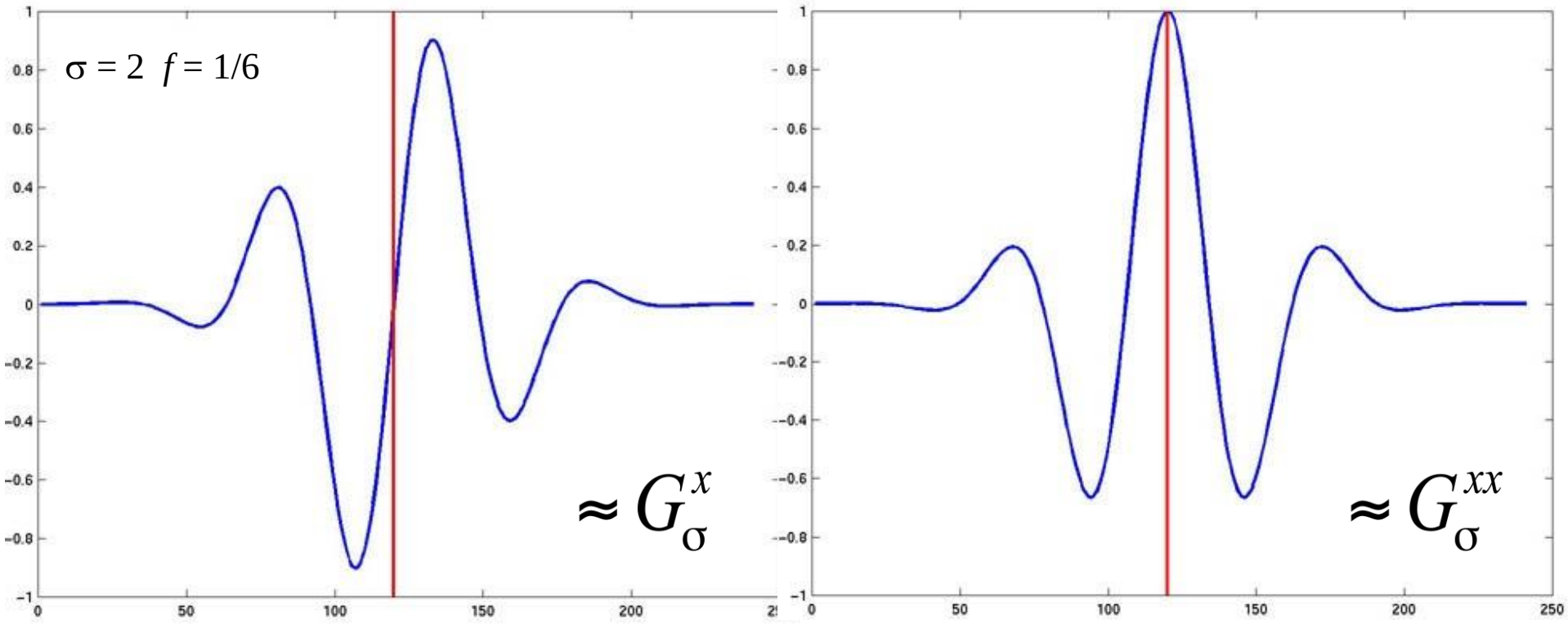
… looks a lot like…



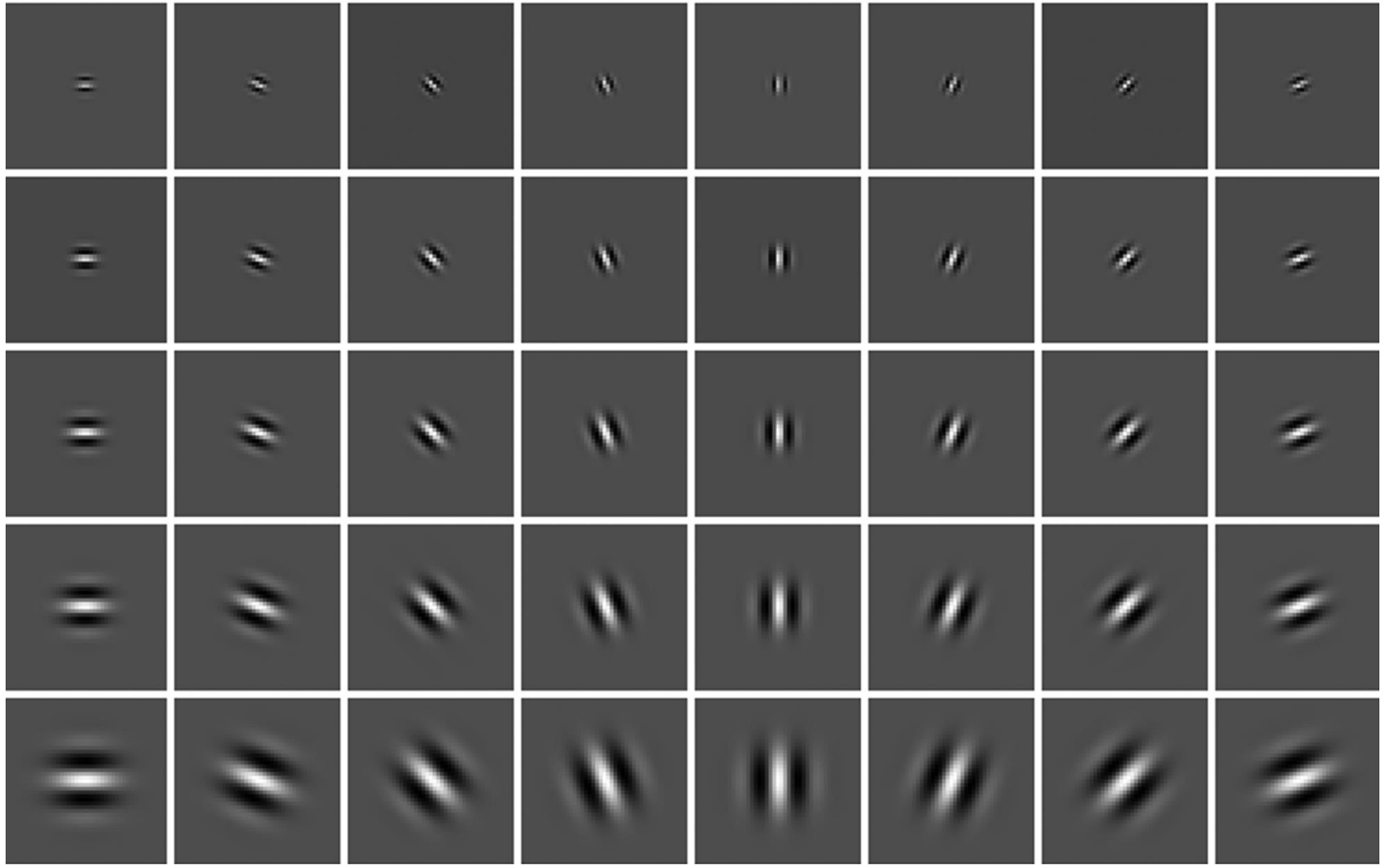
Laplacian



If scale small compared to inverse frequency, the Gabor filters become derivative operators



Directional edge detectors

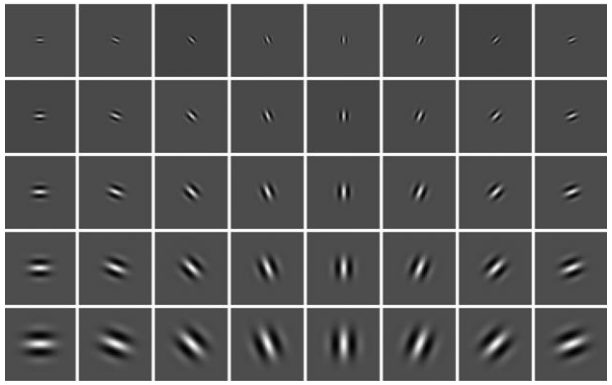


GIST

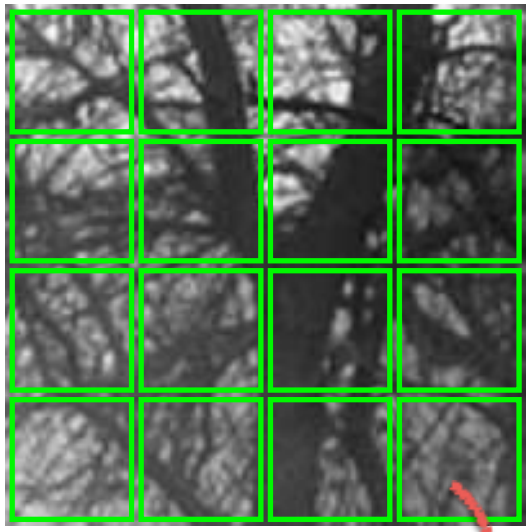
- Compute filter responses (filter bank of Gabor filters)
- Divide image patch into 4×4 cells
- Compute filter response averages for each cell
- Size of descriptor is $4 \times 4 \times N$, where N is the size of the filter bank

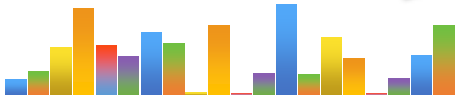
What is the GIST descriptor encoding?

Rough spatial distribution of image gradients



4 x 4 cell





averaged filter responses

Histograms of Oriented Gradients for Human Detection

Navneet Dalal and Bill Triggs

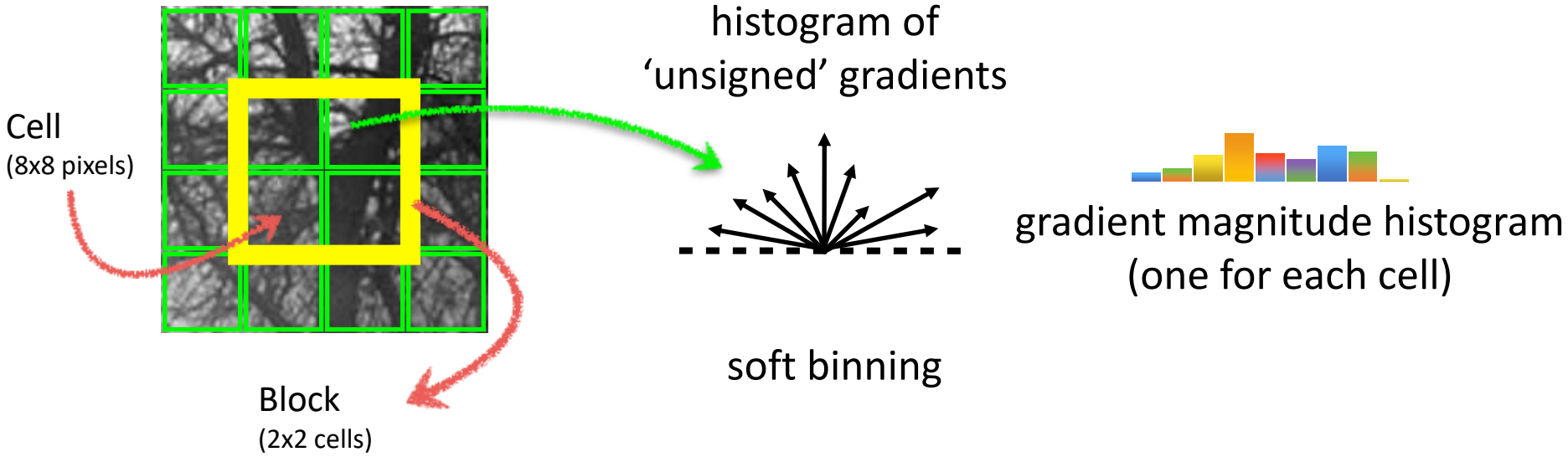
INRIA Rhône-Alps, 655 avenue de l'Europe, Montbonnot 38334, France

{Navneet.Dalal,Bill.Triggs}@inrialpes.fr, http://lear.inrialpes.fr

CVPR 2005

HOG

Dalal, Triggs. **Histograms of Oriented Gradients** for Human Detection. CVPR, 2005



Concatenate and L-2 normalization

Single scale, no dominant orientation

HOG

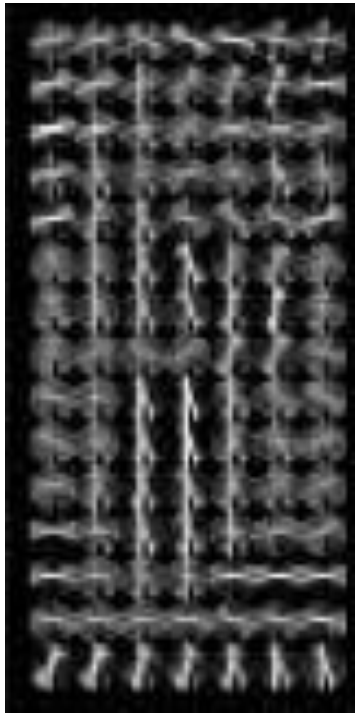
128 pixels
16 cells
15 blocks

1 cell step size

64 pixels
8 cells
7 blocks

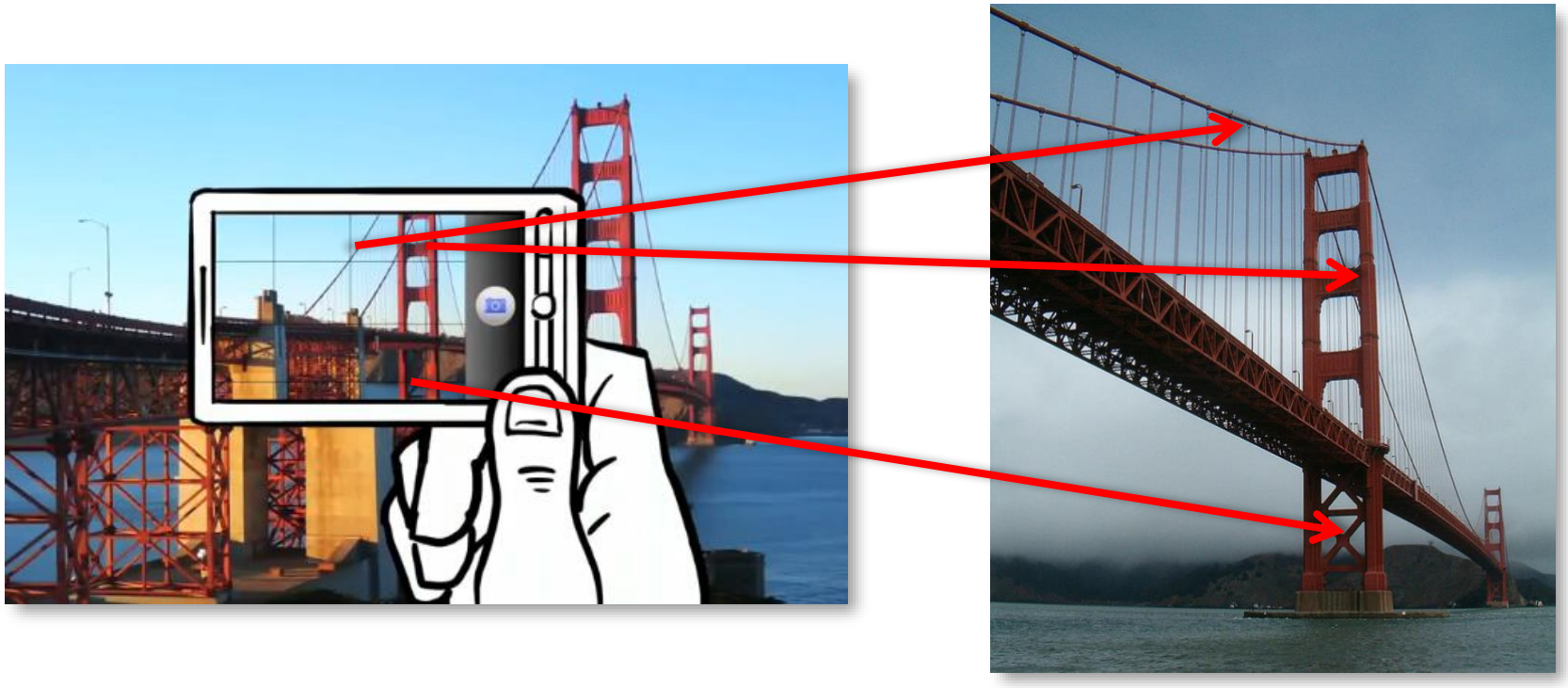
15 x 7 x 4 x 9 = 3780

visualization



Questions?

Feature matching



Feature matching

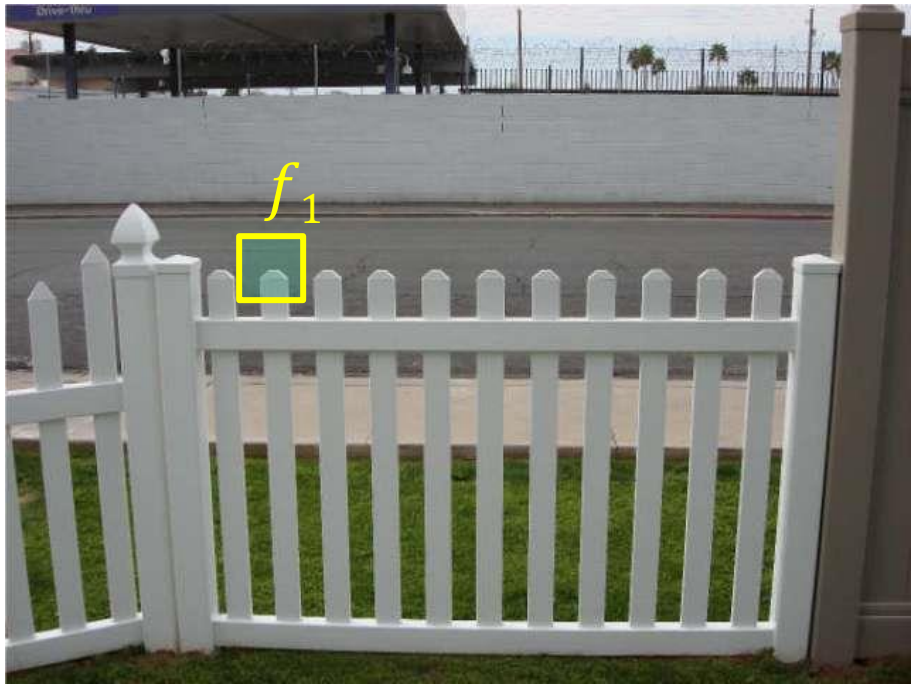
Given a feature in I_1, how to find the best match in I_2?

1. Define distance function that compares two descriptors
2. Test all the features in I_2, find the one with min distance

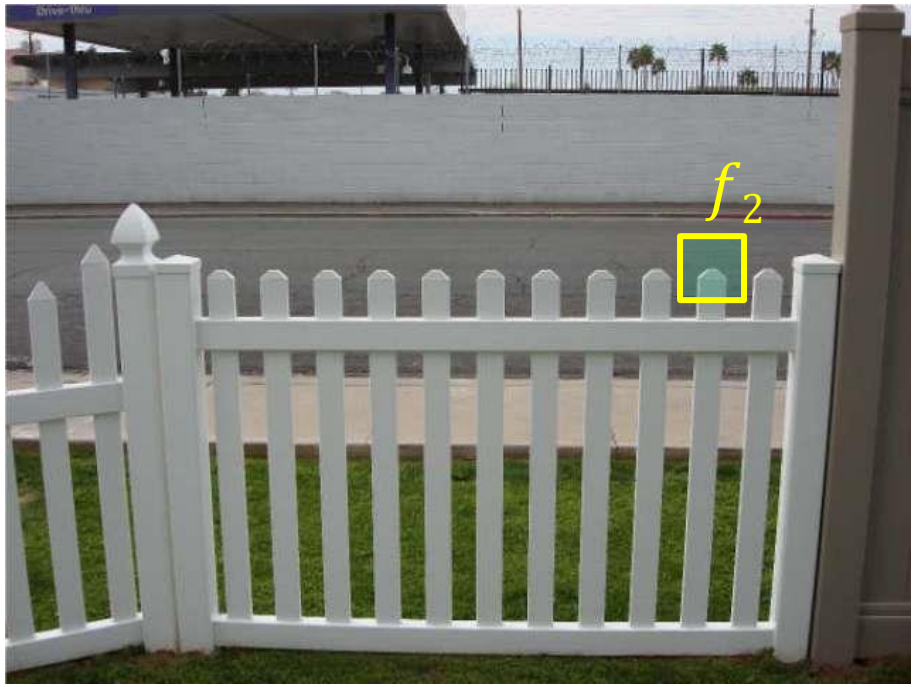
feature distance

How to define the difference between two features f_1, f_2?

- Simple approach: L_2 distance, $||f_1 - f_2||$
- Can give good scores to ambiguous (incorrect) matches



I_1

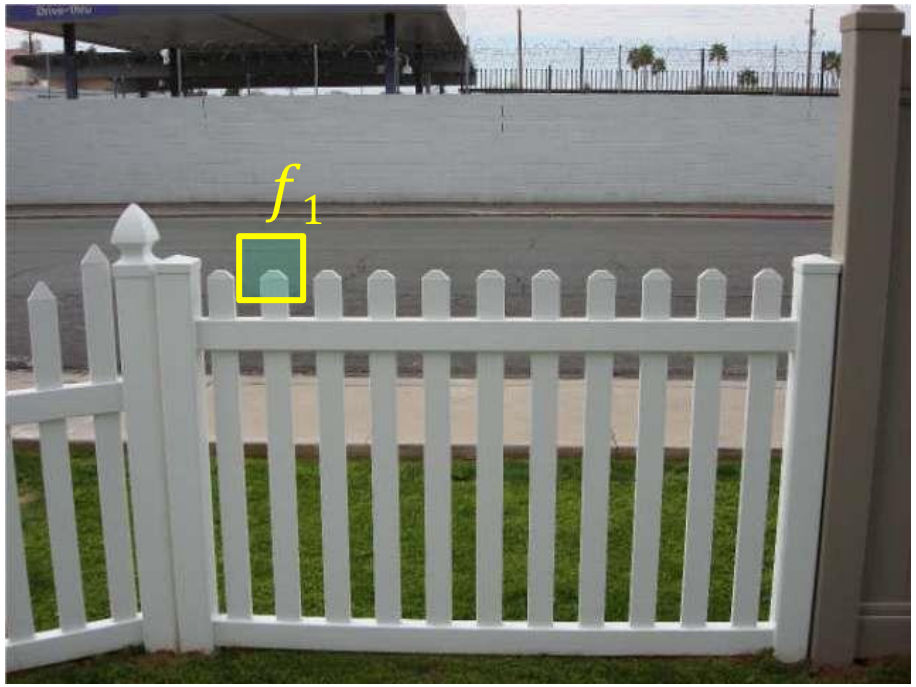


I_2

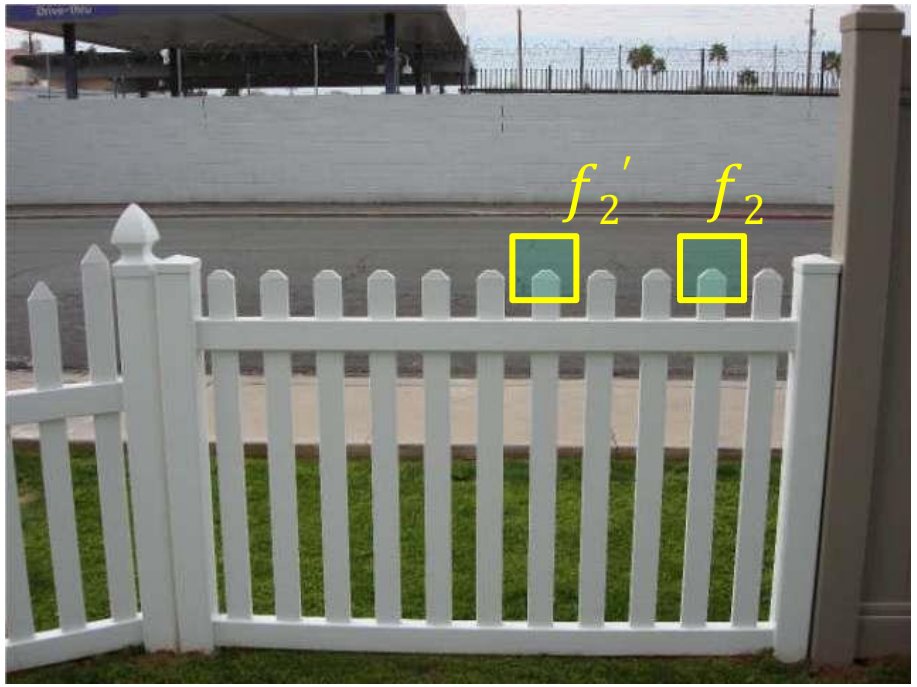
feature distance

How to define the difference between two features f_1, f_2?

- Better approach: ratio distance = $\|f_1 - f_2\| / \|f_1 - f_2'\|$
  - f_2 is best SSD match to f_1 in I_2
  - f_2' is 2nd best SSD match to f_1 in I_2
  - gives large values for ambiguous matches

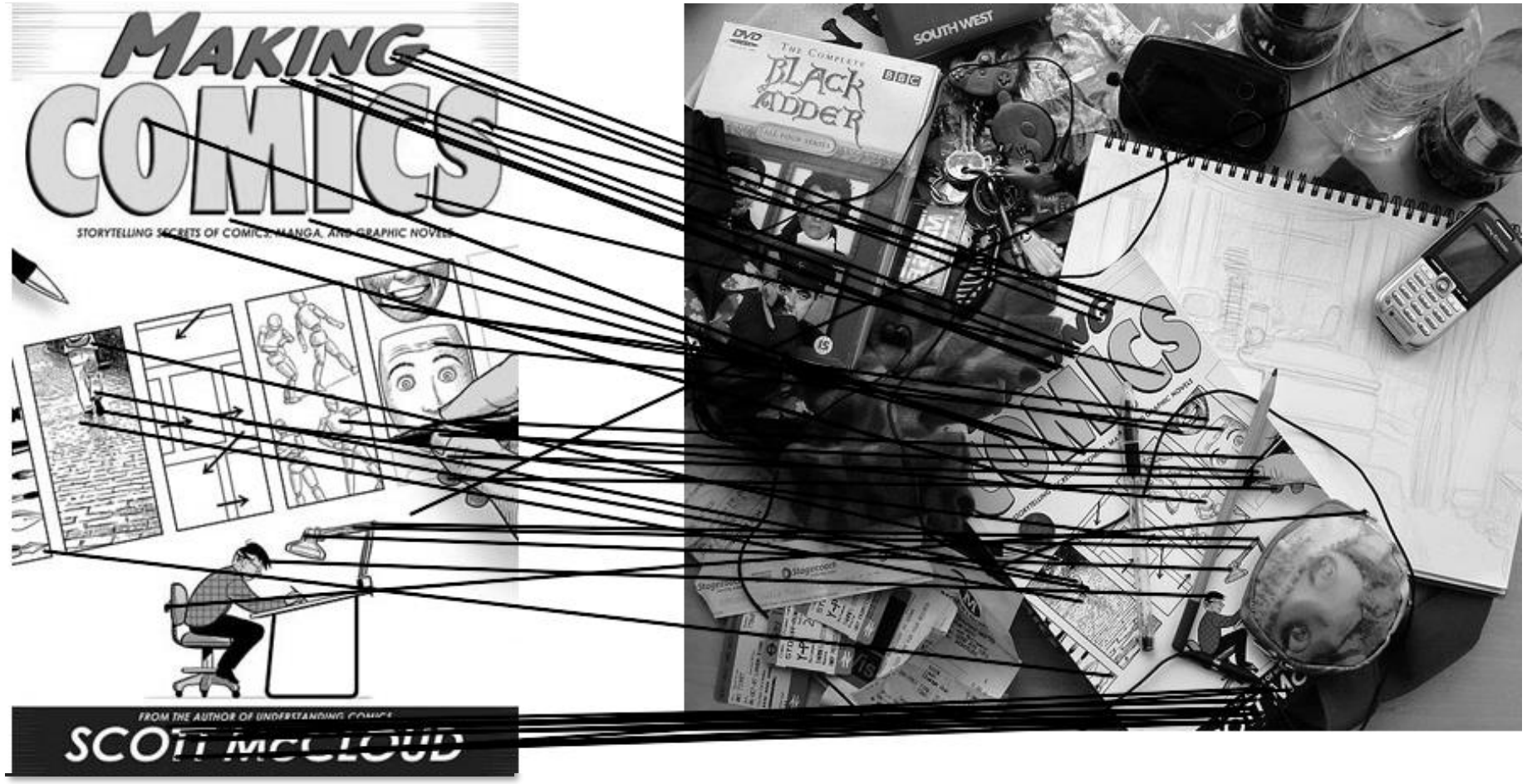


I_1



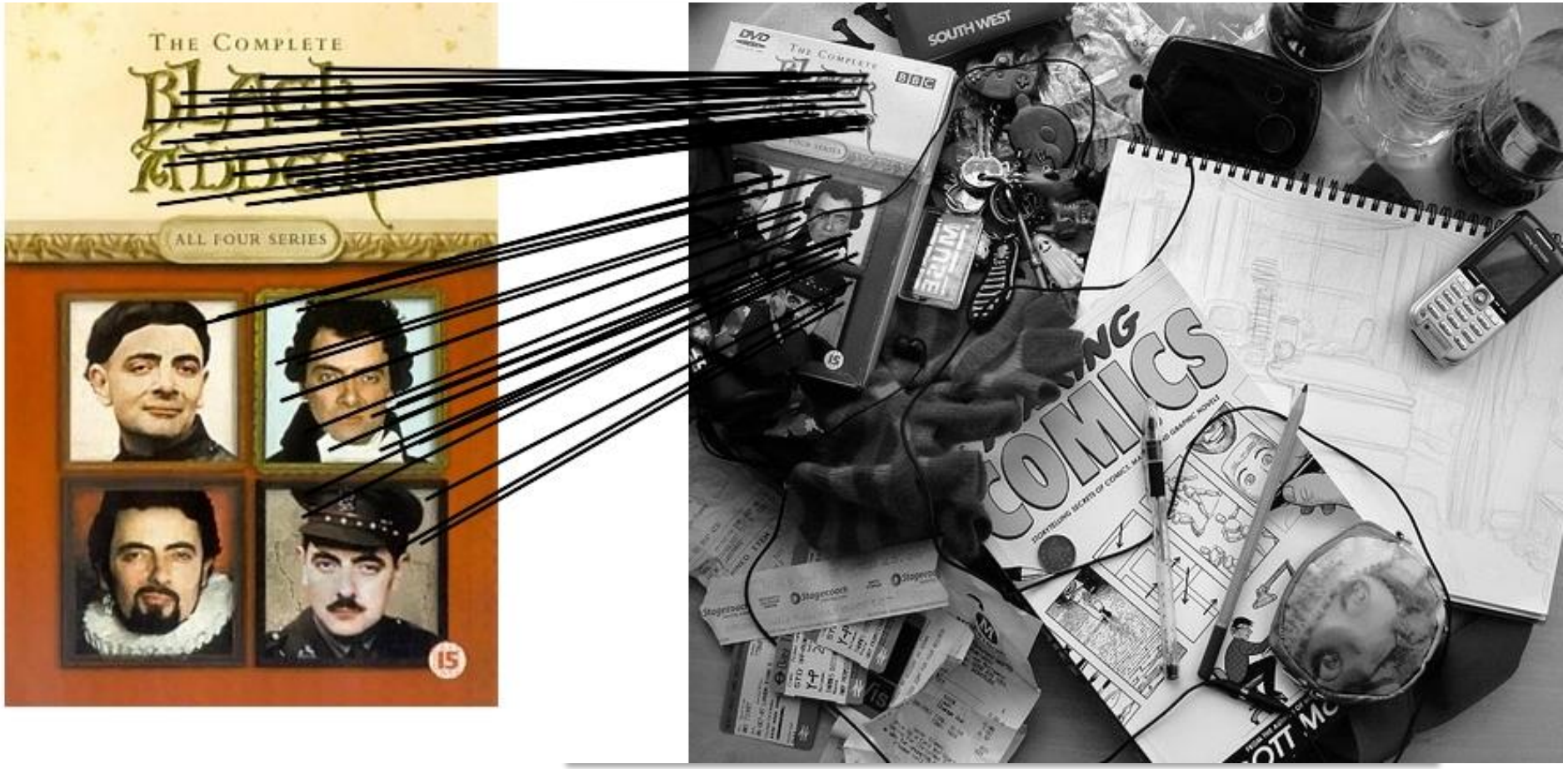
I_2

Feature matching example



51 matches

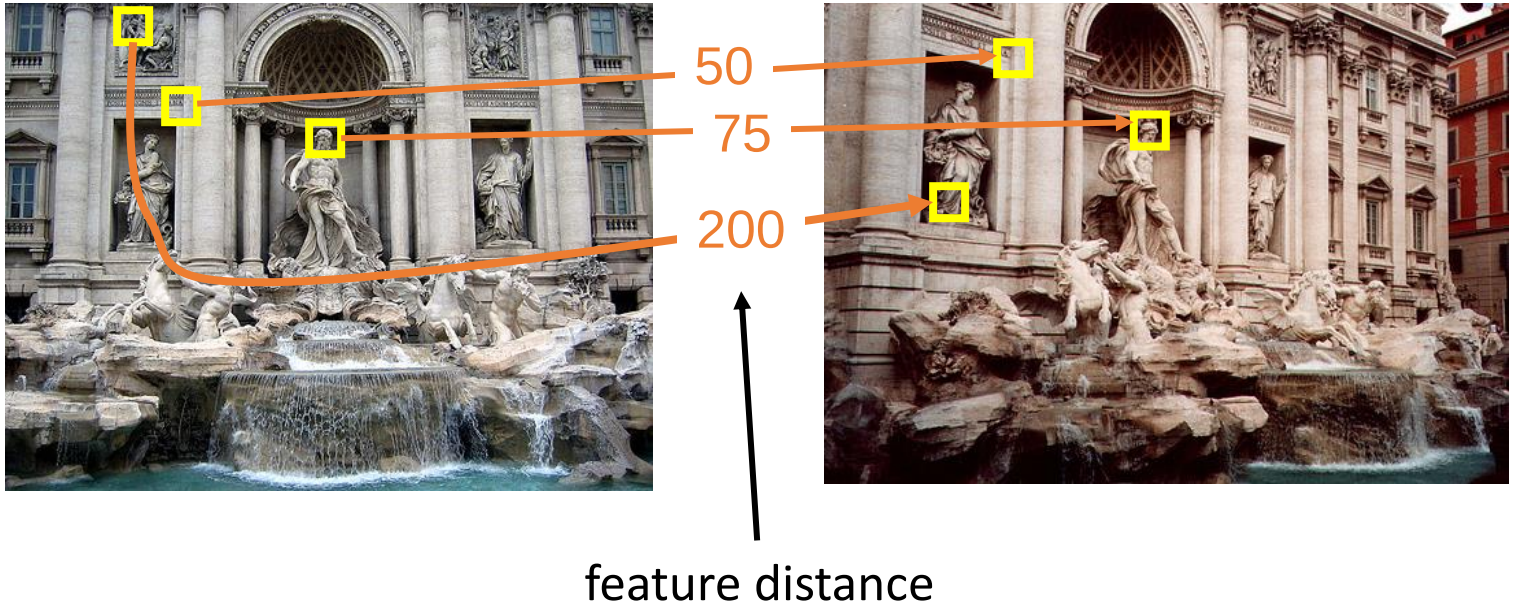
Feature matching example



58 matches

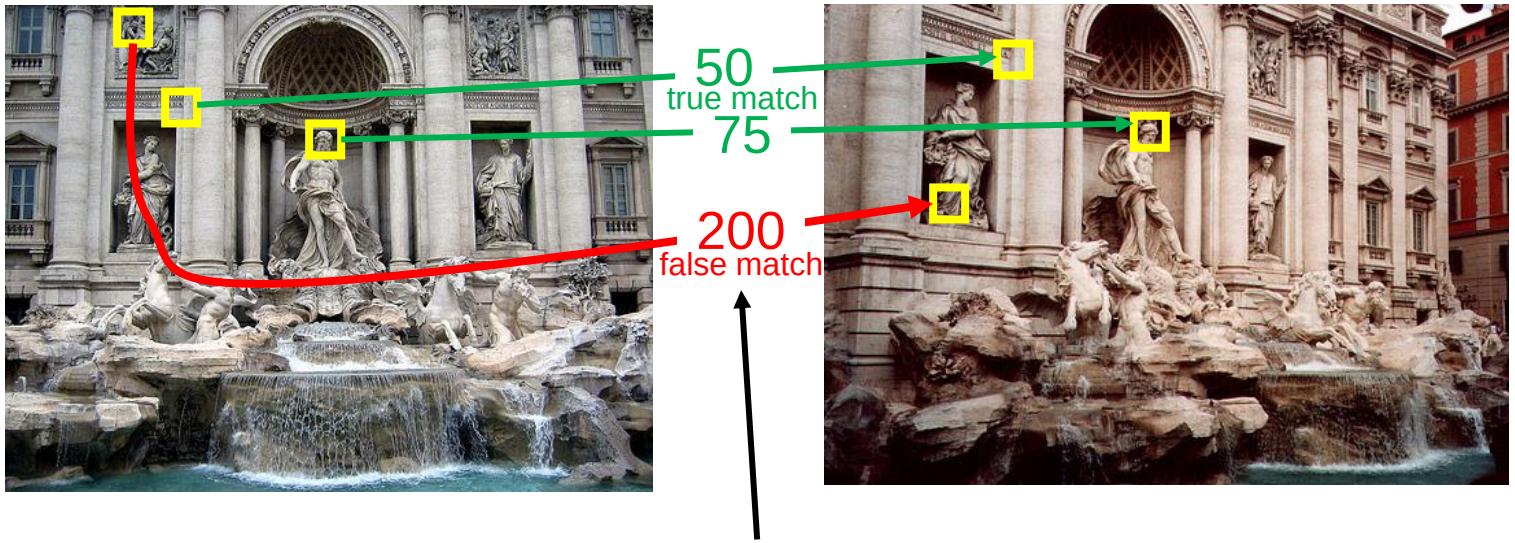
Evaluating the results

How can we measure the performance of a feature matcher?



True/false positives

How can we measure the performance of a feature matcher?



The distance threshold affects performance

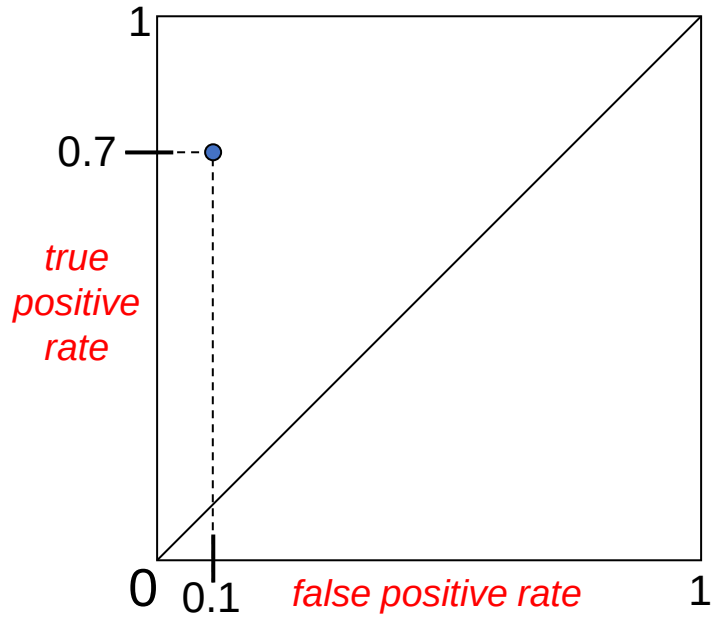
- True positives = # of detected matches that are correct
  - Suppose we want to maximize these—how to choose threshold?
- False positives = # of detected matches that are incorrect
  - Suppose we want to minimize these—how to choose threshold?

Evaluating the results

- How can we measure the performance of a feature matcher?

$$\frac{\text{\# true positives}}{\text{\# GT feature matches}}$$

"recall"



$$\frac{\text{\# false positives}}{\text{\# detected feature matches}}$$

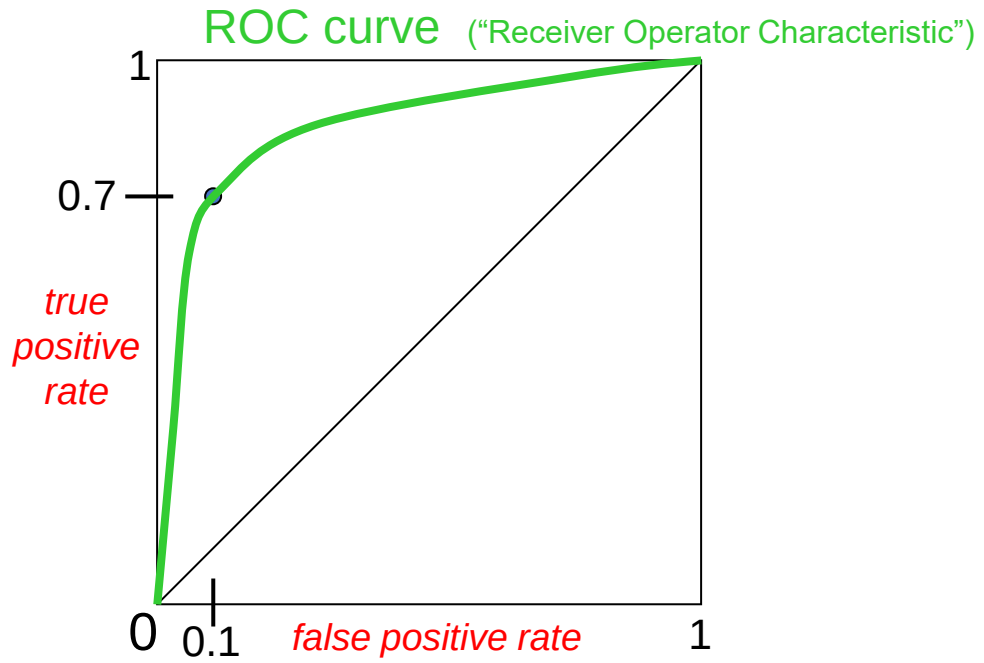
1 - "precision"

Evaluating the results

- How can we measure the performance of a feature matcher?

$$\frac{\text{\# correctly detected matches}}{\text{\# GT feature matches}}$$

"recall"



$$1 - \frac{\text{\# correctly detected matches}}{\text{\# detected feature matches}}$$

1 - "precision"