

Salient Point and Features

Giacomo Boracchi

Image Analysis and Computer Vision, Polimi

December 6th 2022

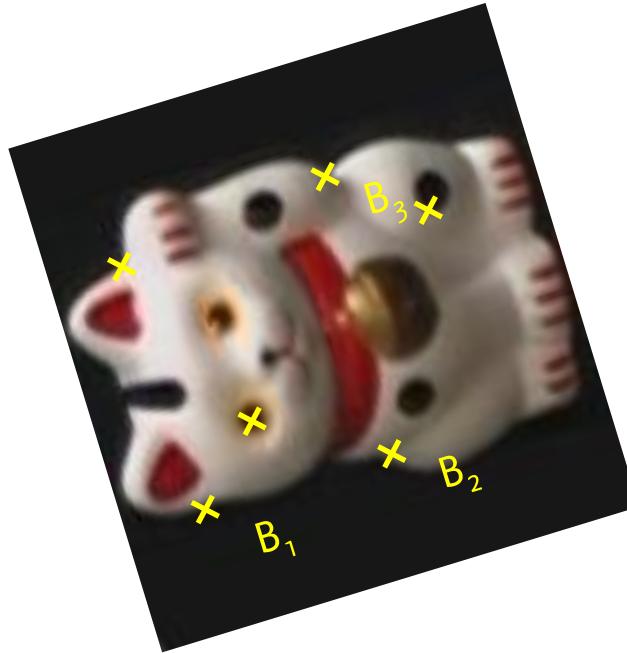
<https://boracchi.faculty.polimi.it/>

giacomo.boracchi@polimi.it

The General Picture

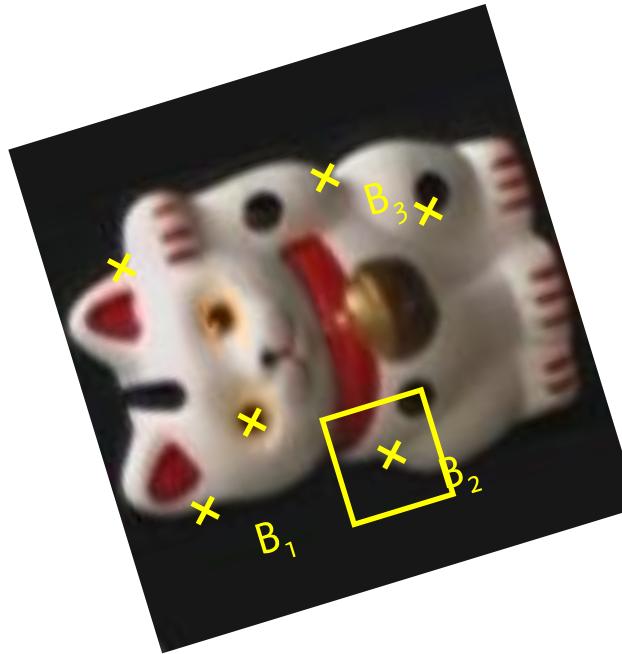


The General Picture



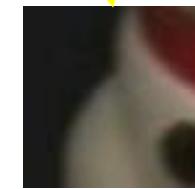
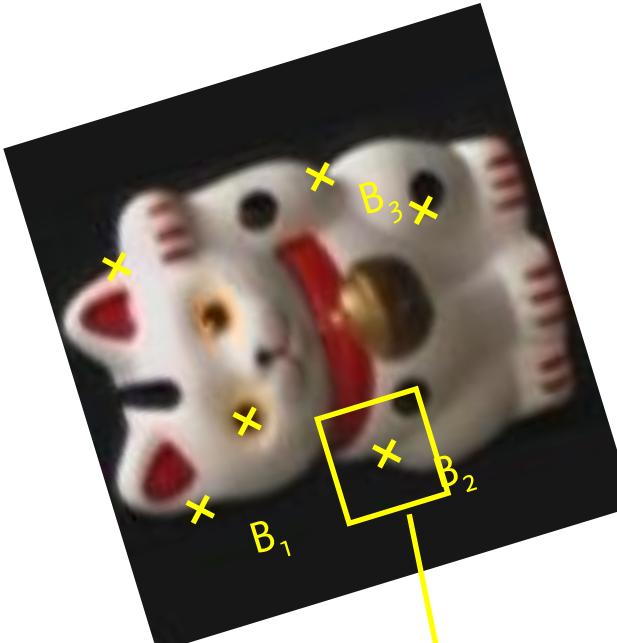
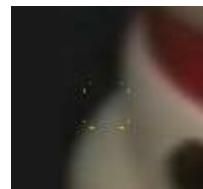
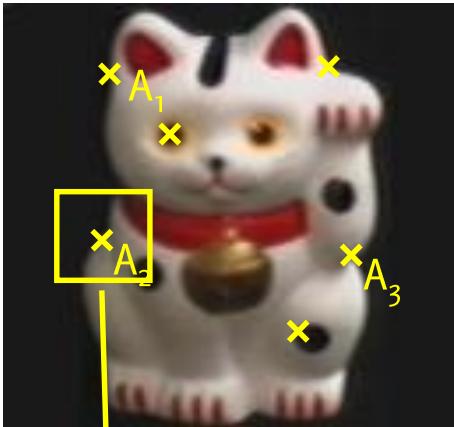
1. Find a set of distinctive key-points

The General Picture



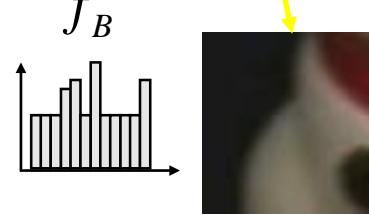
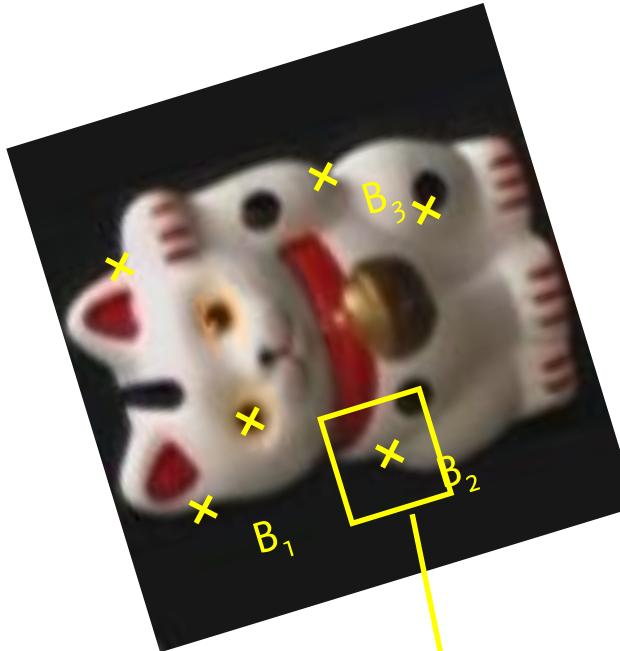
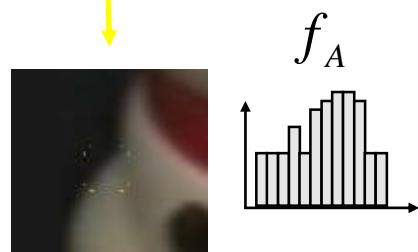
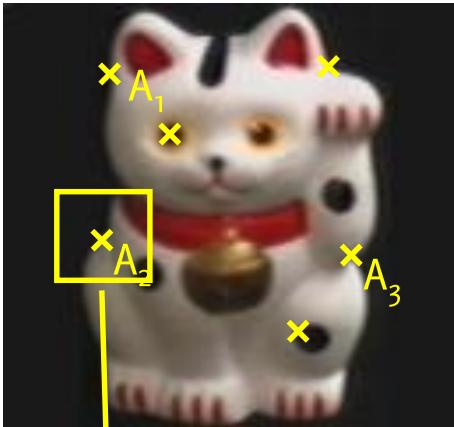
1. Find a set of distinctive keypoints
2. Define a region around each keypoint

The General Picture



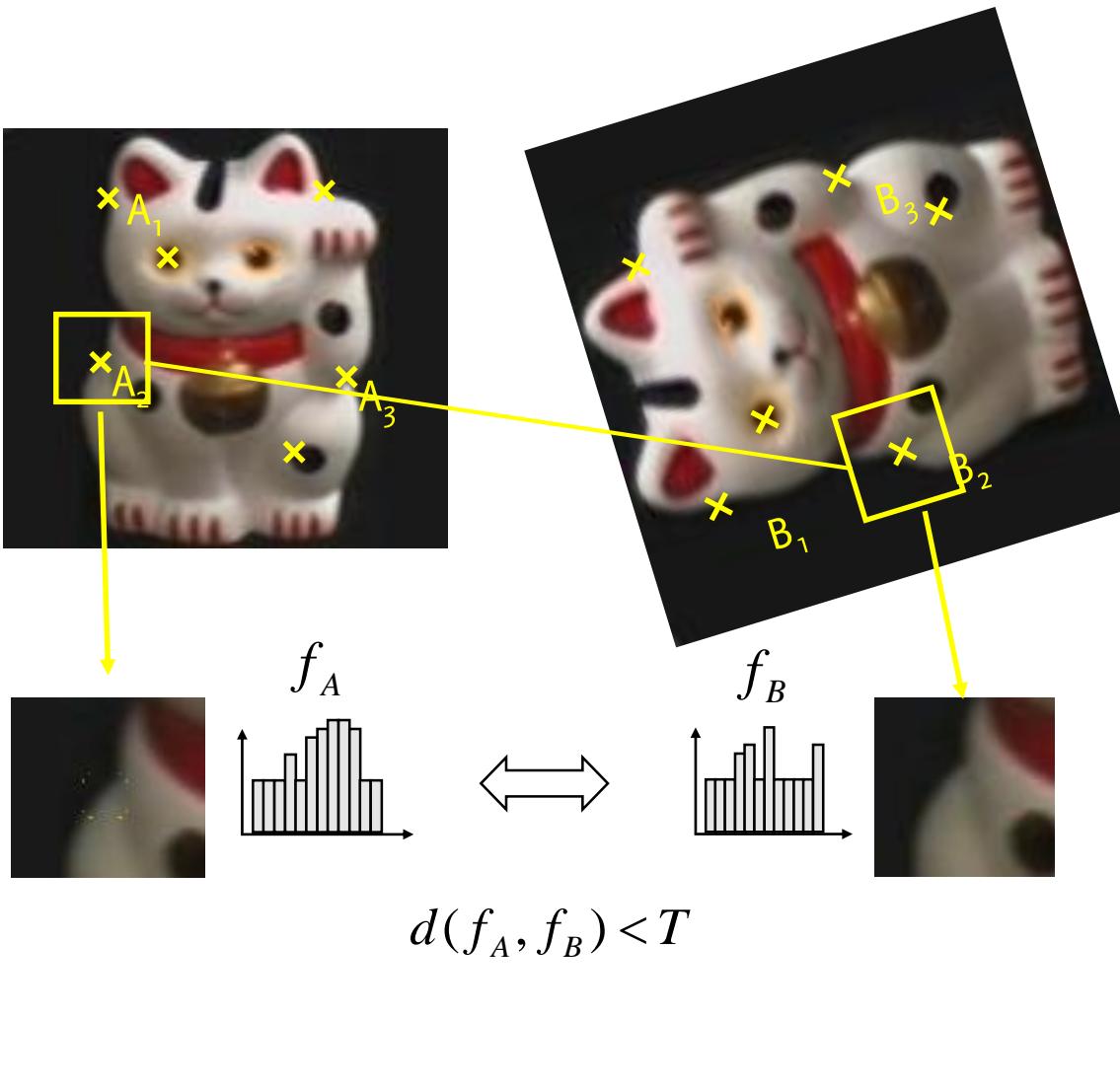
1. Find a set of distinctive keypoints
2. Define a region around each keypoint
3. Extract and normalize the region content

The General Picture



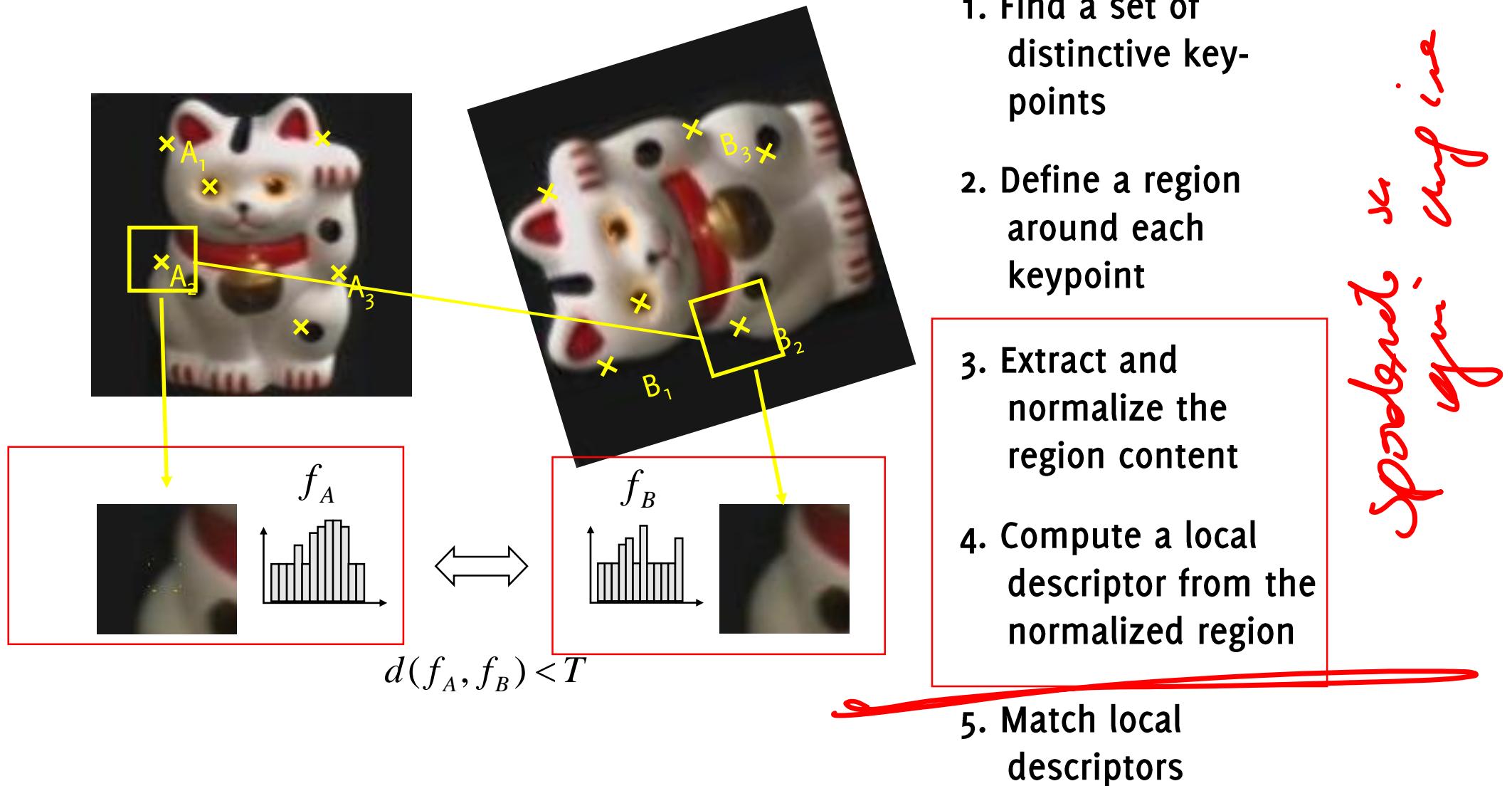
1. Find a set of distinctive keypoints
2. Define a region around each keypoint
3. Extract and normalize the region content
4. Compute a local descriptor from the normalized region

The General Picture

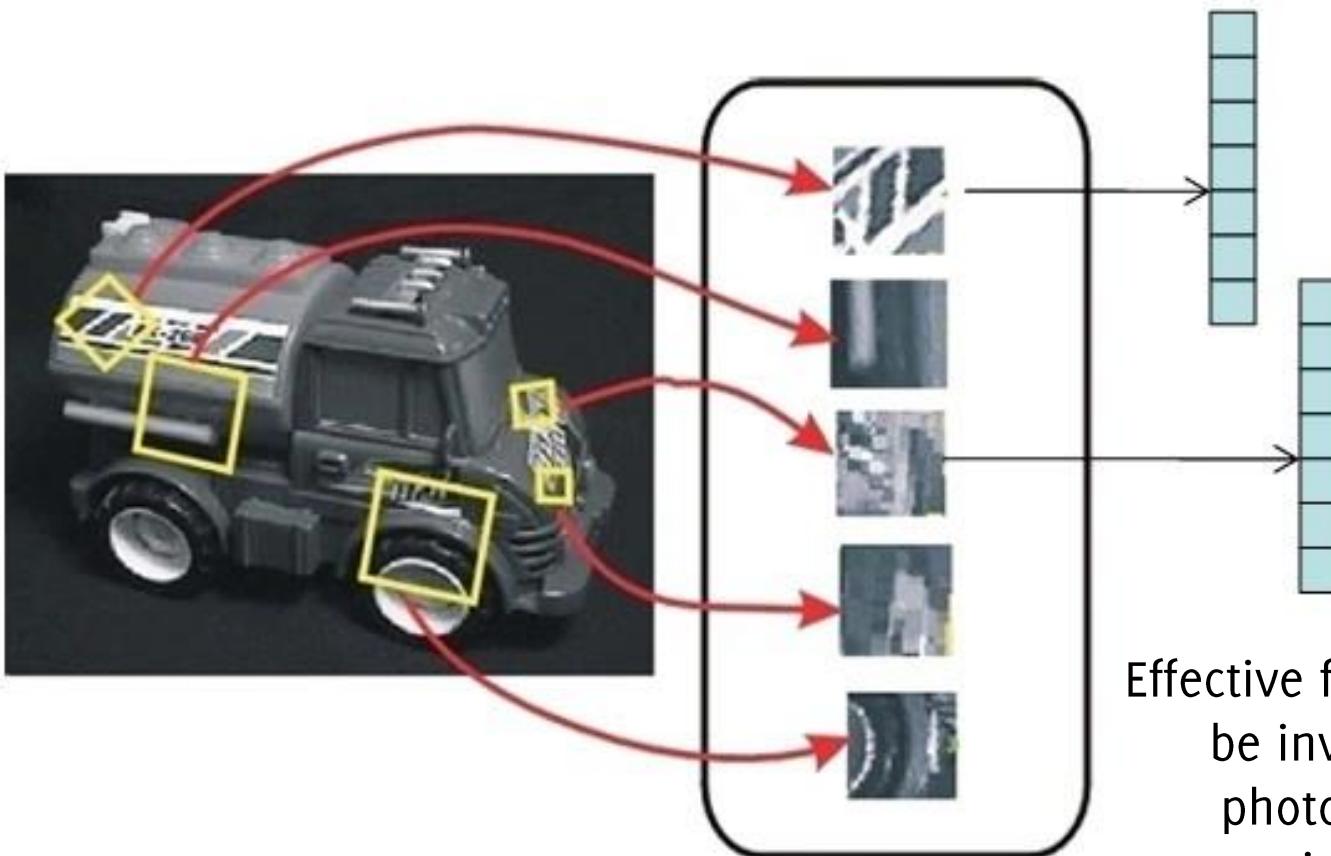


1. Find a set of distinctive keypoints
2. Define a region around each keypoint
3. Extract and normalize the region content
4. Compute a local descriptor from the normalized region
5. Match local descriptors

The General Picture



The General Picture

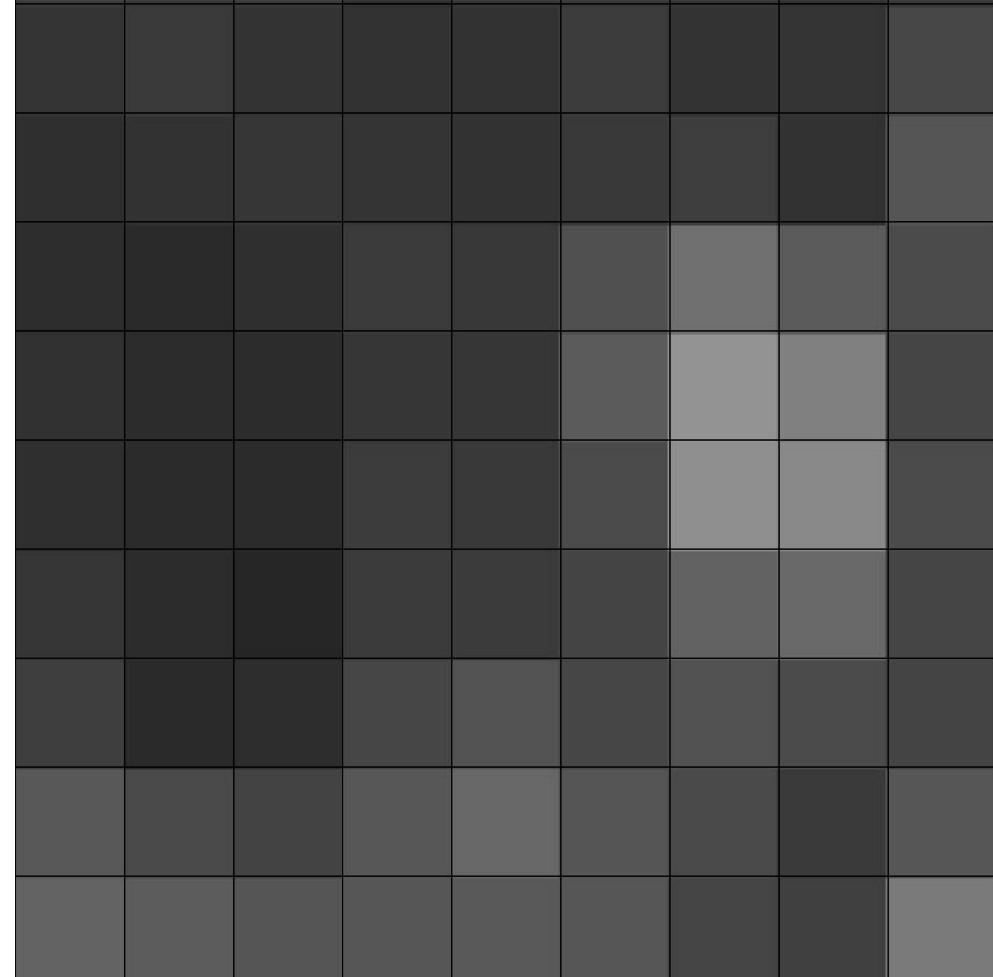


Features can be stacked in vectors

Effective features have to be invariant w.r.t. photometric and geometric transformation of the image

Features and Keypoints

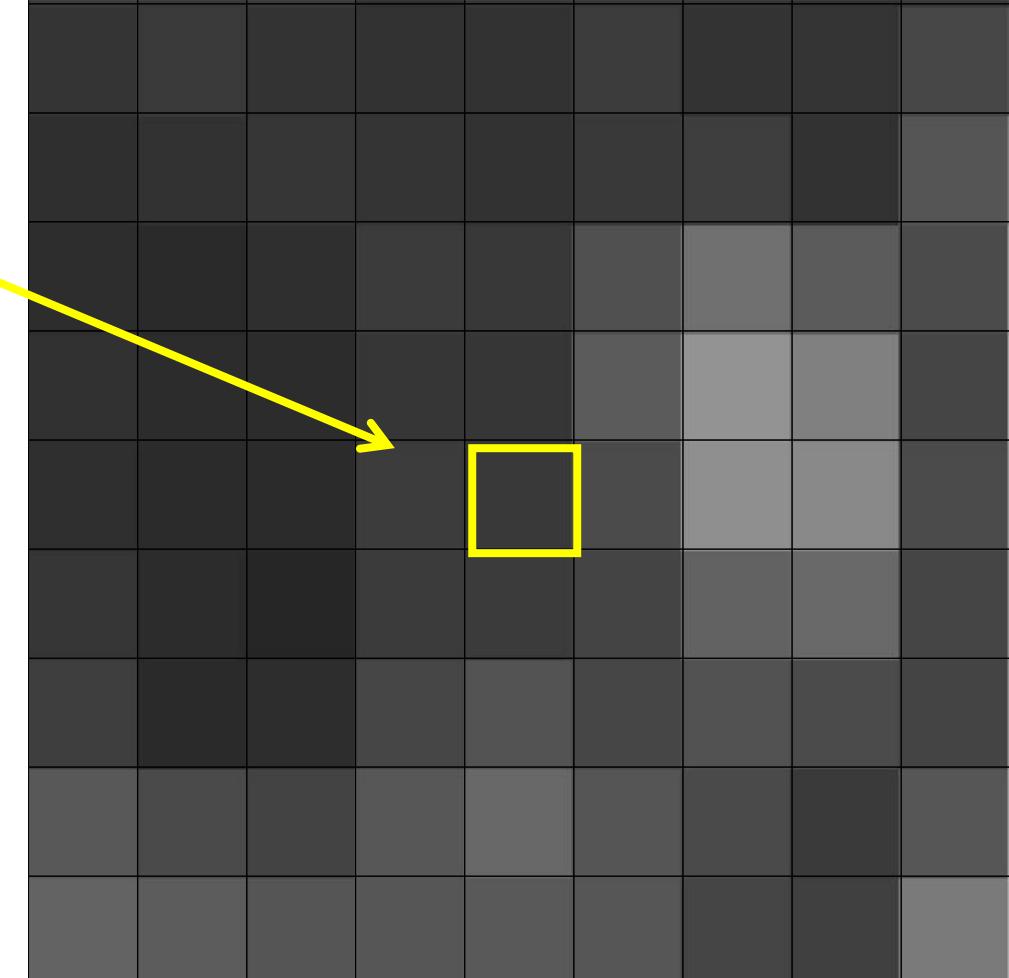
Consider an Image Patch



Features and Keypoints

Consider an Image Patch

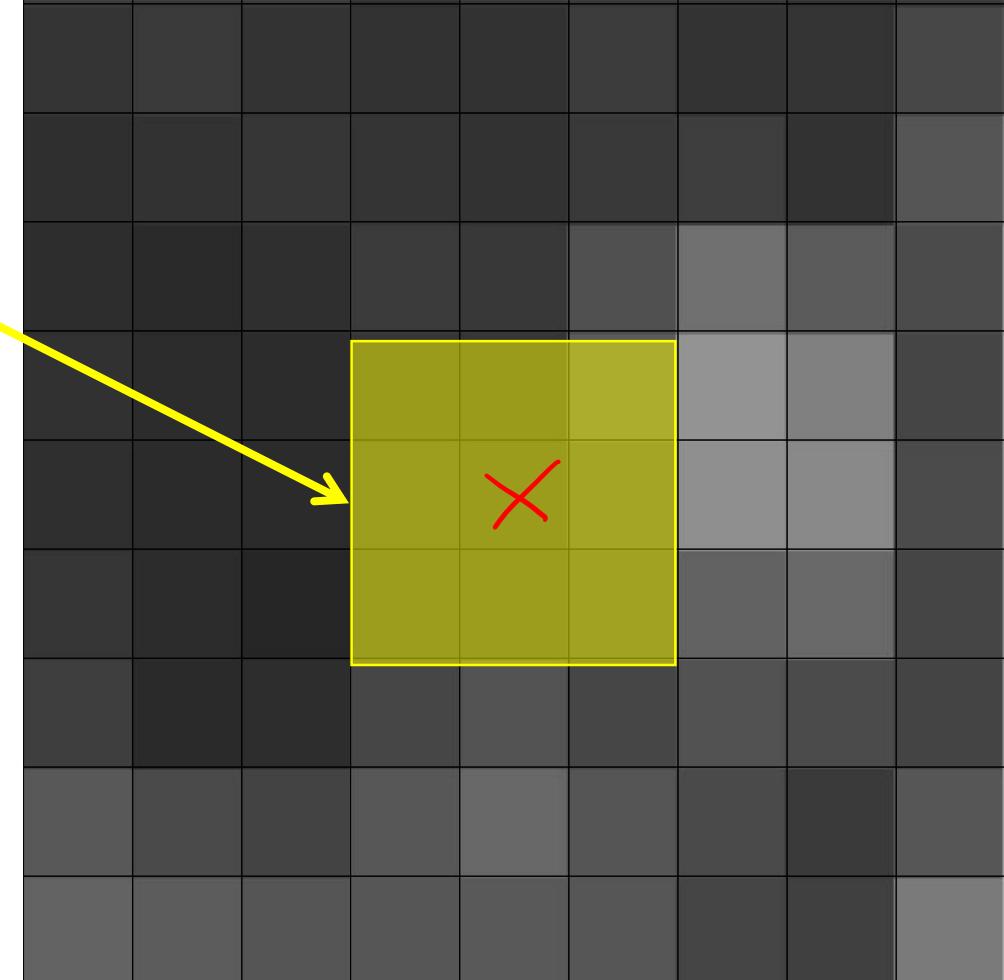
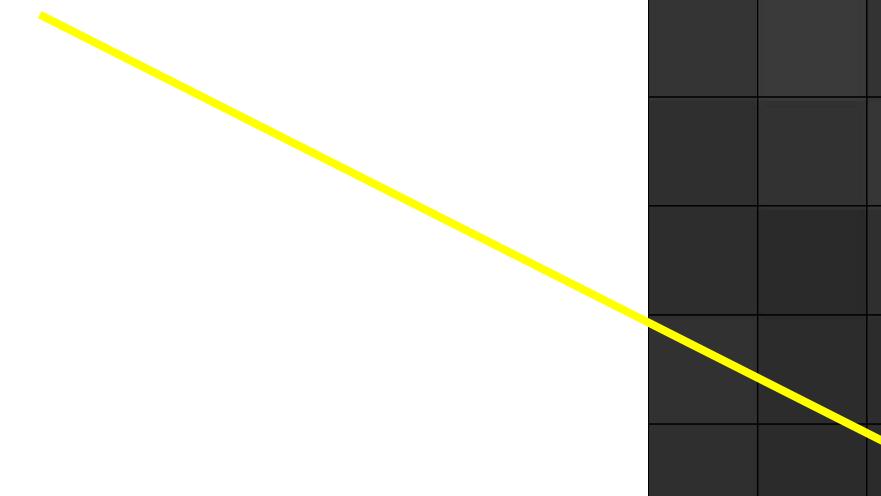
Keypoint: The coordinates of a point where the image content is sort of relevant



Features and Keypoints

A **Feature** could be

- an Keypoint neighbor

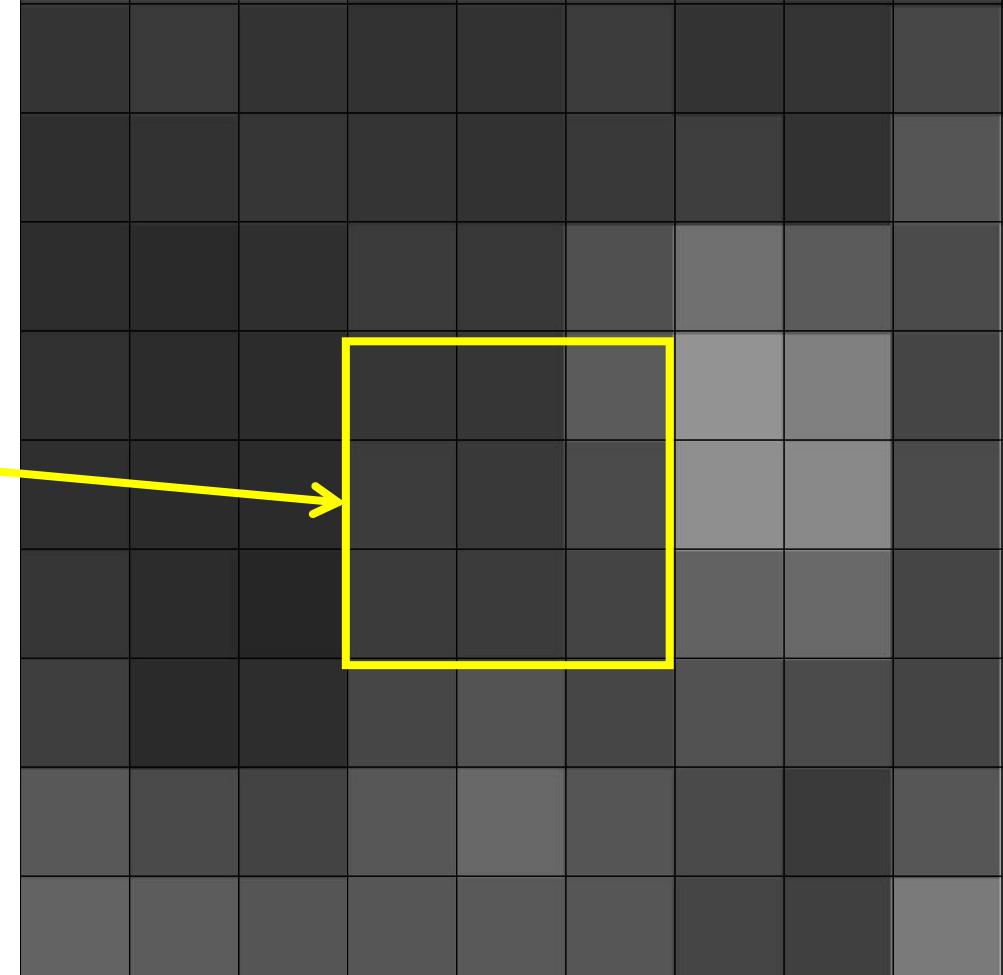


Features and Keypoints

A **Feature** could be

- an Keypoint neighbor
- some measures computed in an image neighbor:
 - mean
 - variance
 - principal directions
 - ...

stacked in a vector, thus yielding a descriptor



Object Recognition by Computer Vision Features

Keypoint detection: identifying coordinates where the image is considered meaningful for addressing some task

Design Criteria: Keypoints have to be repeatable



Object Recognition by Computer Vision Features

Descriptor computation: compute a vector that describes the content of an image in a region around the keypoint

Design Criteria: Features have to be stable

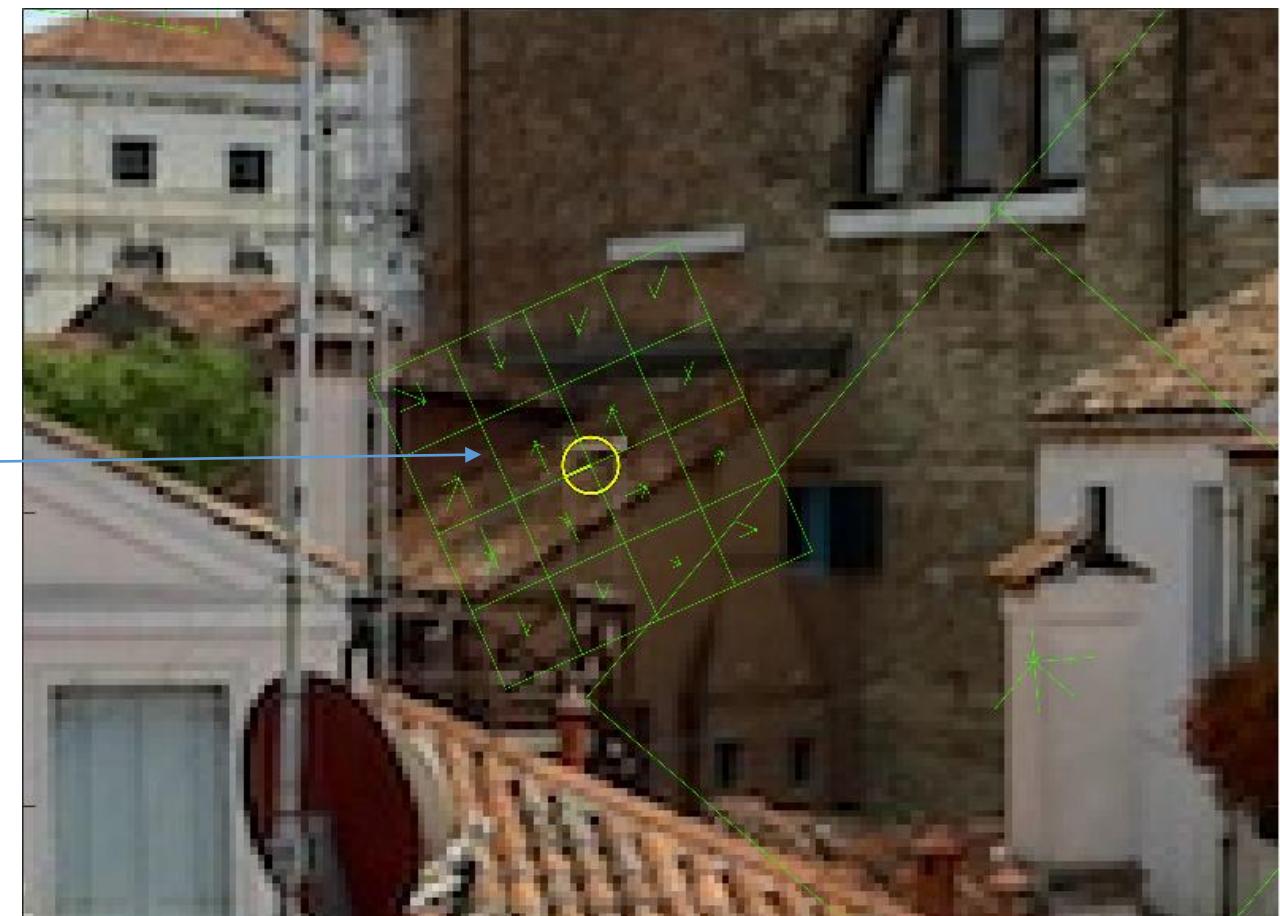


Object Recognition by Computer Vision Features

Descriptor computation: compute a vector that describes the content of an image in a region around the keypoint

Design Criteria: Features have to be stable

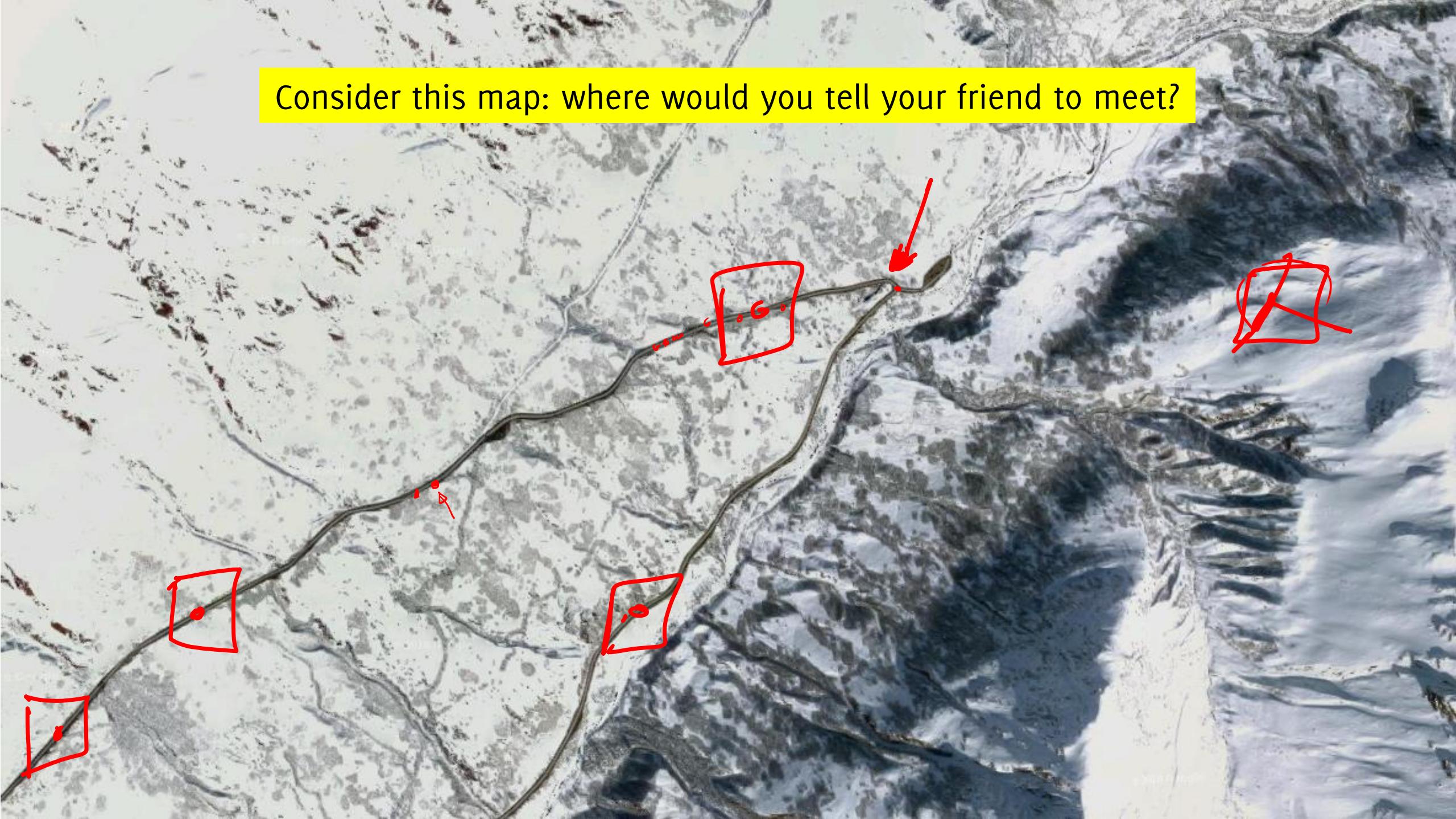
There is a vector in \mathbb{R}^{128}
associated to each of
these regions



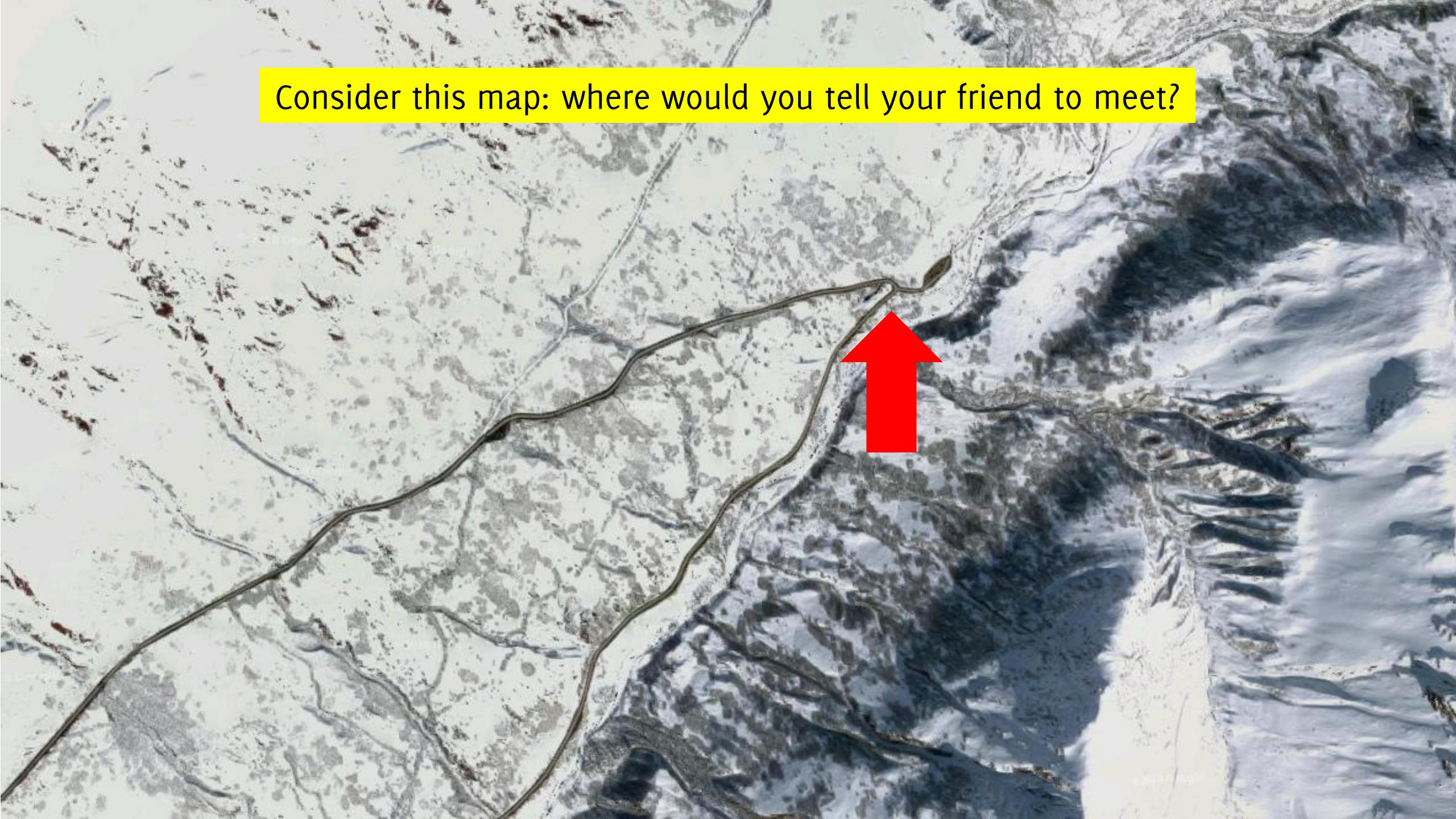
Keypoint Detection: The Rationale

The principle underpinning many corner detection algorithms

Consider this map: where would you tell your friend to meet?



Consider this map: where would you tell your friend to meet?



Keypoint Properties

Keypoints are expected to be in regions where the image is:

- **Well-defined**: i.e. distinctive, neighboring points should all be different.
- **Stable** across views: same scene point should be extracted when the viewpoint slightly changes

These are necessary properties to achieve **repeatable keypoints**

Keypoint Detection

A point is *interesting* when the image content around there is dissimilar from the neighboring ones.

- We need a measure to assess local similarity / dissimilarity in images

The typical figures of merit to extract keypoints are:

- Gradient Based (ex Harris, Hessian)
- Phase Based (Kovesi)
- Entropy Based (Zisserman)

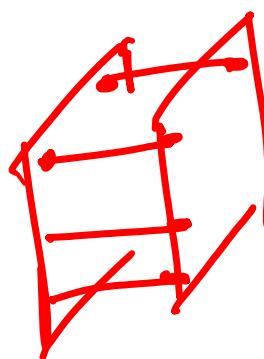
and the Keypoints are located as local maxima of these figure of merit over the whole image.

Comparing image regions

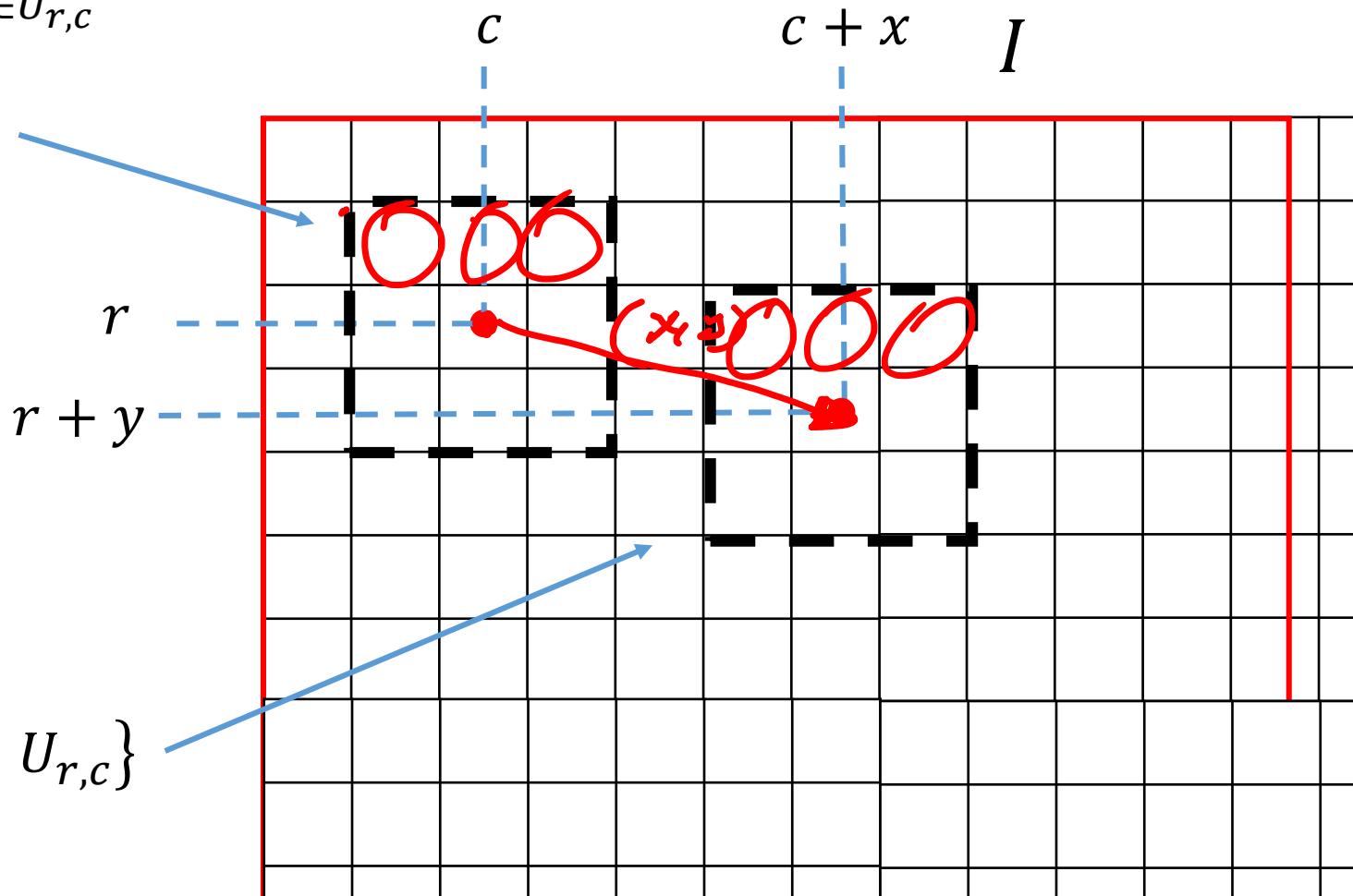
Dissimilarity Measure: the Sum of Squared Distances (SSD)

$$\boxed{E_{x,y}(r,c)} = \sum_{(u,v) \in U_{r,c}} [I(u,v) - I(u-x, v-y)]^2$$

$$\{I(u,v), (u,v) \in U_{r,c}\}$$



$$\{I(u-x, v-y), (u,v) \in U_{r,c}\}$$



Let's go back to our image to better understand SSD



The SSD as the norm of a vector

The SSD is the ℓ^2 norm of the vector given by the difference of two image patches

$$E_{x,y}(r, c) = \left\| \begin{array}{c} \text{[Image Patch]} - \\ \text{[Image Patch]} \end{array} \right\|_2^2$$

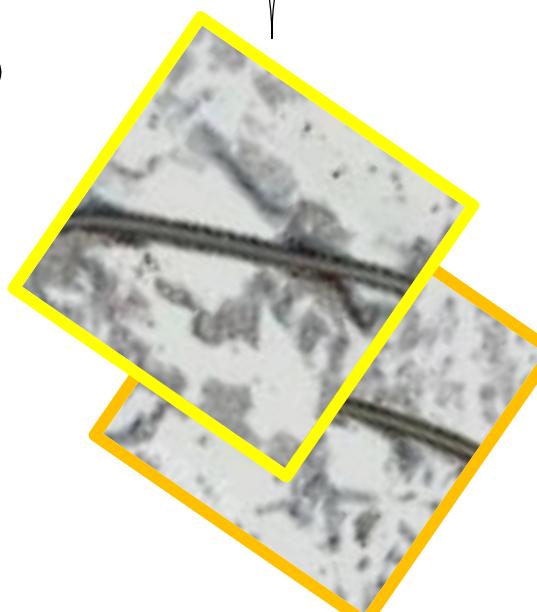
The diagram illustrates the computation of the Sum of Squared Differences (SSD). It shows two image patches, each highlighted with a yellow border. These patches are subtracted from each other, indicated by a minus sign between them. The result of this subtraction is a difference patch, also highlighted with a yellow border. This difference patch is then squared (indicated by a red '2' above the vertical bar) and summed to find the ℓ^2 norm, which represents the SSD.

The SSD as the norm of a vector

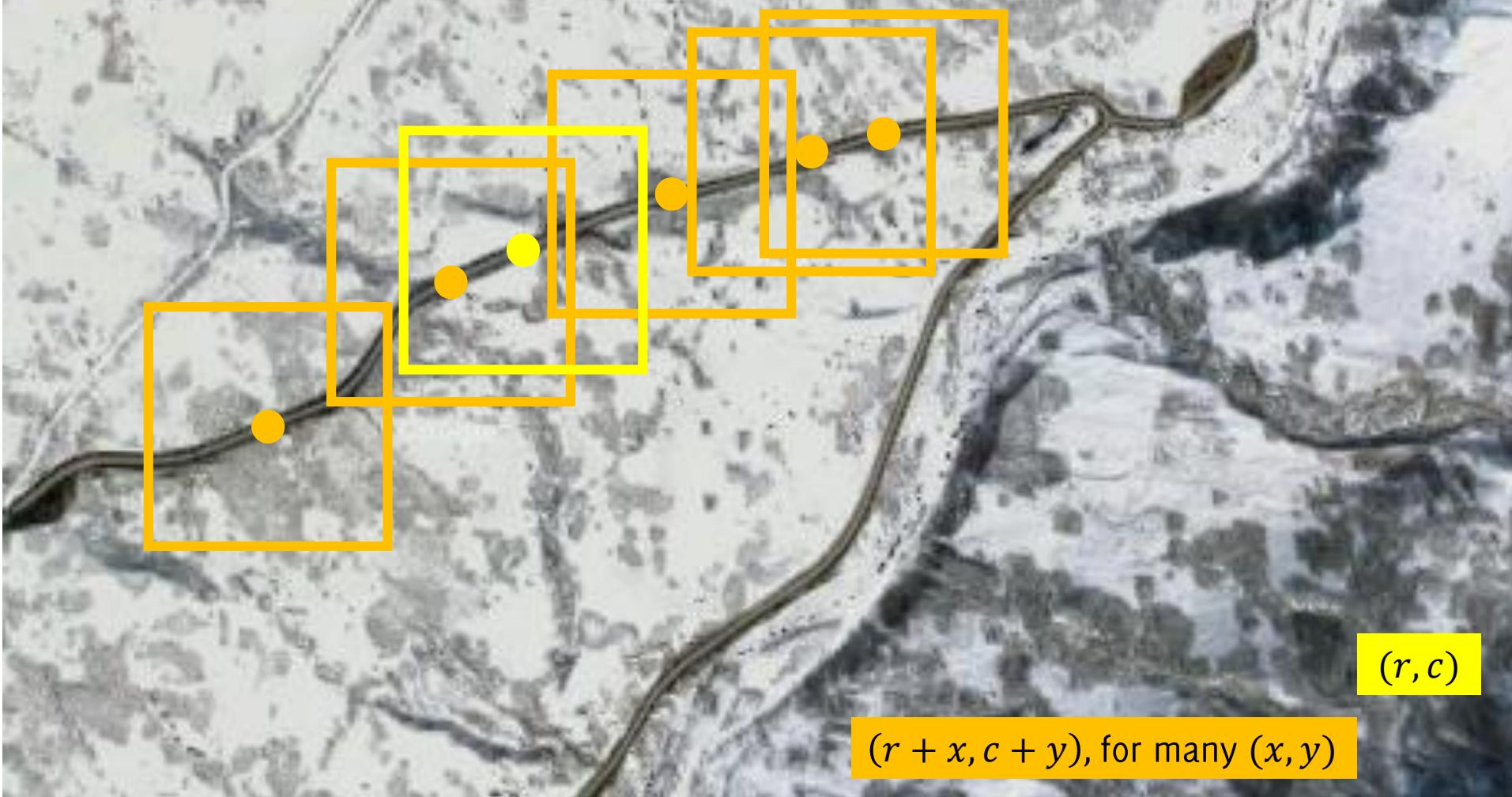
The SSD is the ℓ^2 norm of the vector given by the difference of two image patches

$$E_{x,y}(r, c) = \left\| \begin{array}{c} \text{[Image patch]} \\ - \\ \text{[Image patch]} \end{array} \right\|_2$$

The pixel-wise difference among these two patches is likely to be very close to zero



Let's go back to our image to better understand SSD



There exist many values (x, y) yielding a patch centered in $(r + x, c + y)$ that is very similar to the one centered in (r, c)

The SSD as the norm of a vector

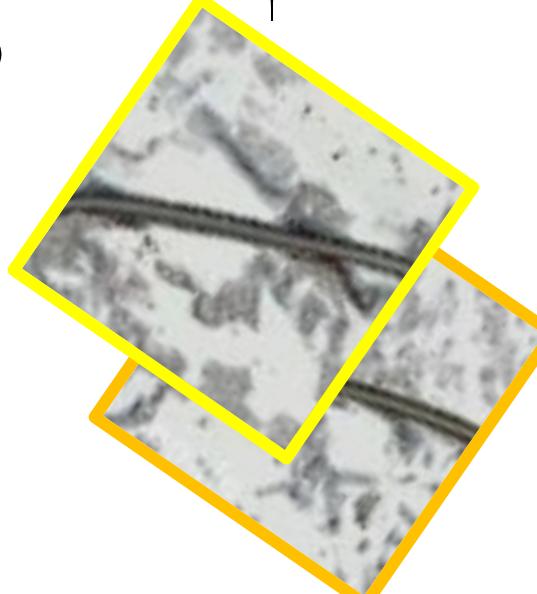
The SSD is the ℓ^2 norm of the vector given by the difference of two image patches

$$E_{x,y}(r, c) = \left\| \begin{array}{c} \text{[Image patch]} \\ - \\ \text{[Image patch]} \end{array} \right\|_2$$

The pixel-wise difference among these two patches is likely to be very close to zero

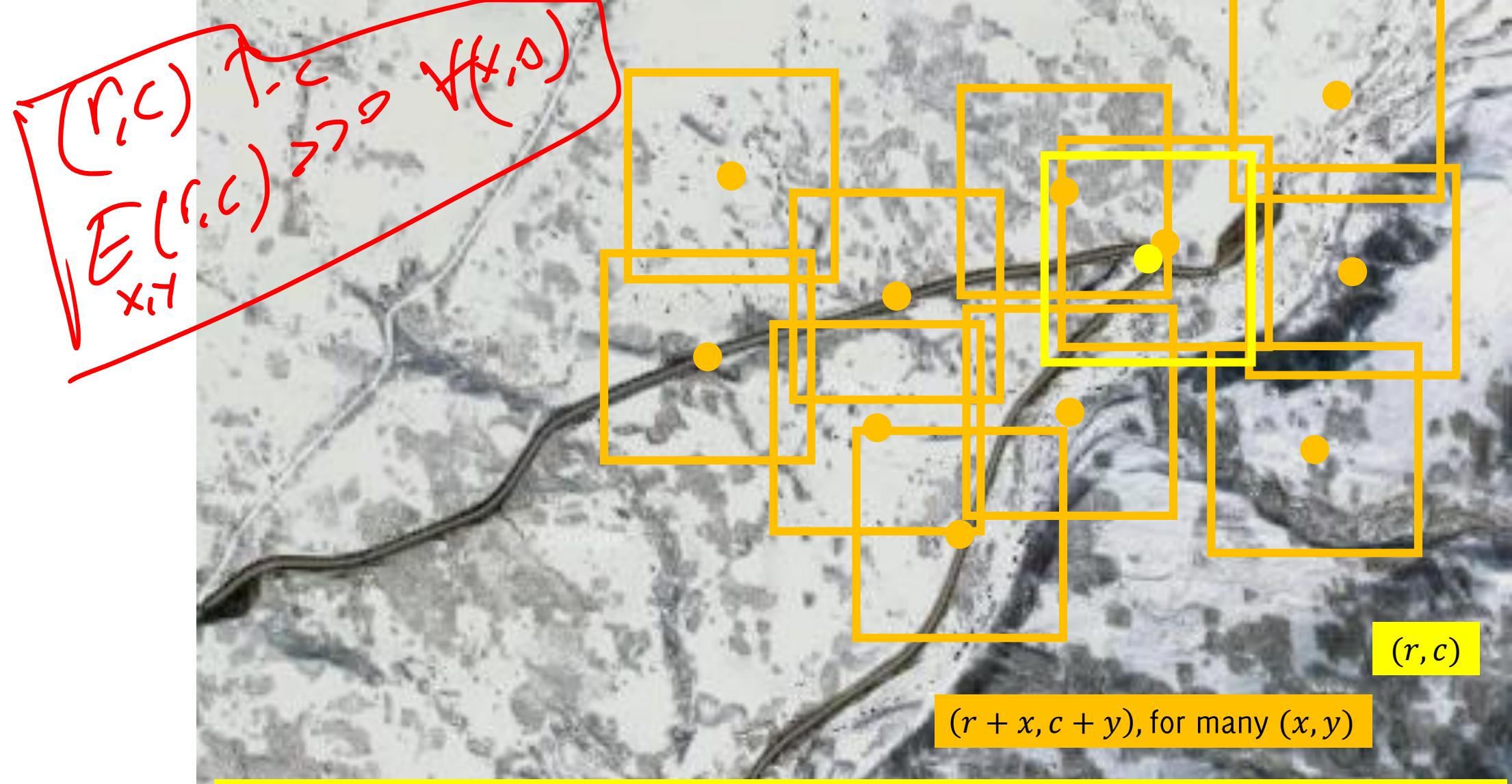
..the same holds for many orange alternatives centered in $(r + x, c + y)$ along that road.

Thus, (r, c) is not a keypoint

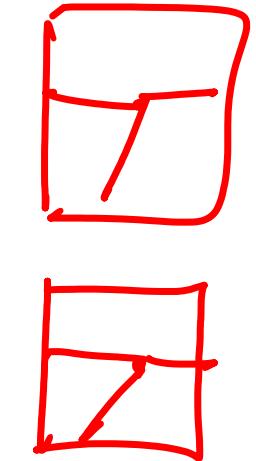




But there are locations (r, c) yielding a patch that is very different from all the neighboring ones, thus which is almost unique.



But there are locations (r, c) yielding a patch that is very different from all the neighboring ones, thus which is almost unique.
These are the locations we want to select as keypoints



The SSD as the norm of a vector

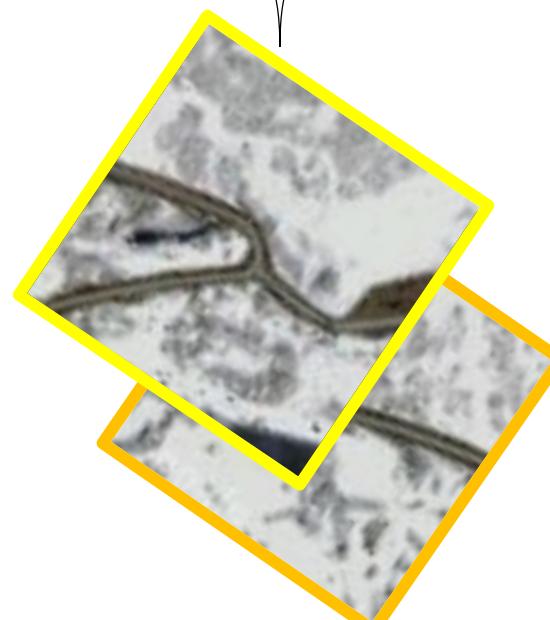
The SSD is the ℓ^2 norm of the vector given by the difference of two image patches

$$E_{x,y}(r, c) = \left\| \begin{array}{c} \text{yellow patch} \\ - \\ \text{orange patch} \end{array} \right\|_2$$

If the pixel-wise difference among

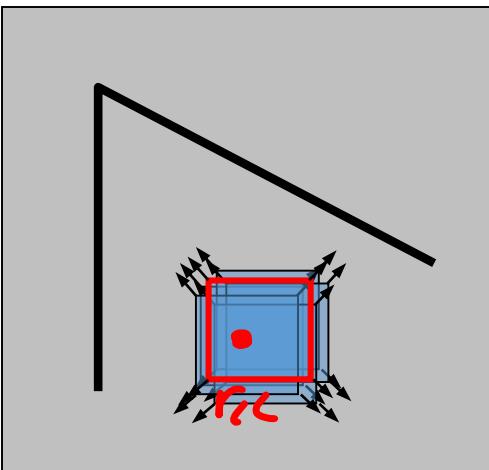
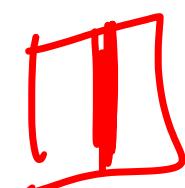
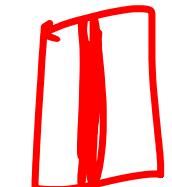
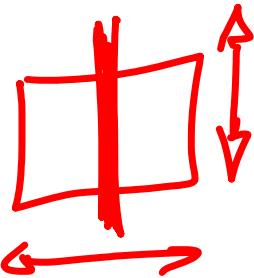
- the yellow patch in (r, c) and
- any orange alternative in patch $(r + x, c + y)$

is likely to be large, then (r, c) is then a good candidate for becoming a keypoint

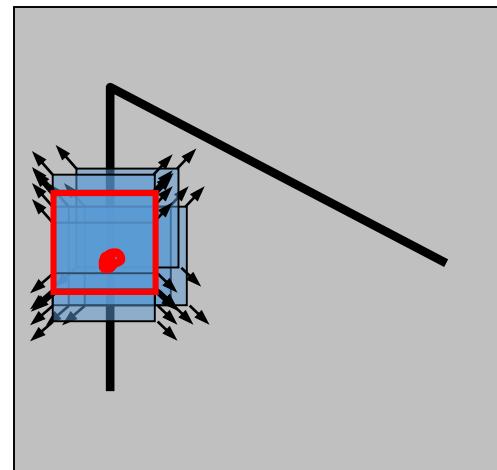


The rationale behind many corner detectors

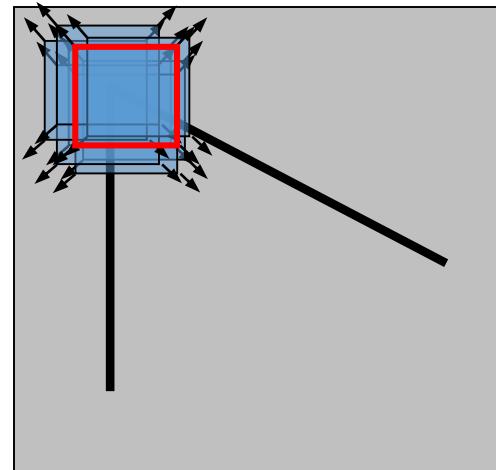
Compute the Sum of Square Distances between the image values on the green square at different position



“flat” $E(x)$:
no change in
all directions



“edge”:
no change along
the edge direction



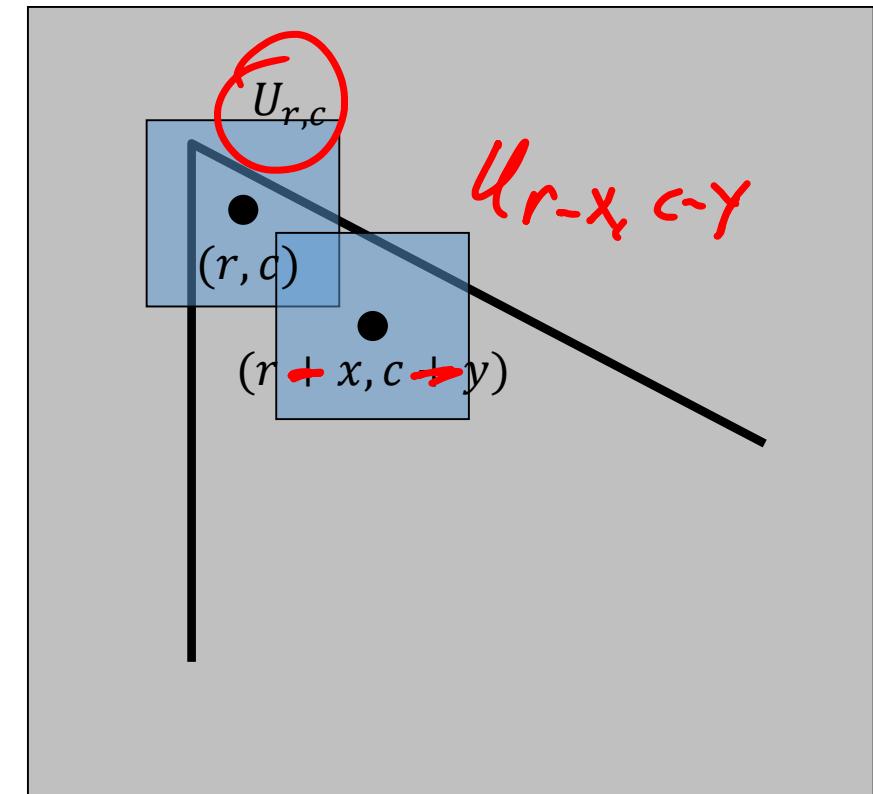
“corner”:
significant change
in all directions

Keypoint Detection: Harris Corner

A meaningful example to be found in many other
algorithms

Setting up the stage

- (r, c) point where to compute the output response (candidate keypoint)
- $U_{r,c}$ neighborhood identifying the blue area
- $E_{x,y}(r, c)$ difference between the green square centered in (r, c) and the square centered in $(r - x, c - y)$
- The pixels inside $U_{r,c}$ are indexed by (u, v)



Moreavec (80) – Corner Detection

Corner measure as the SSD over a fixed set of displacements

$$E_{x,y}(r,c) = \sum_{(u,v) \in U_{r,c}} w_{r,c}(u,v)[I(u,v) - I(u-x, v-y)]^2$$
$$(x,y) \in \{(1,0), (0,1), (-1,0), (0,-1)\}$$

$w_{r,c}$ is a window centered in (r,c) , which defines each pixel neighbor
 $U_{r,c}$ (e.g., the green square in the previous slides)

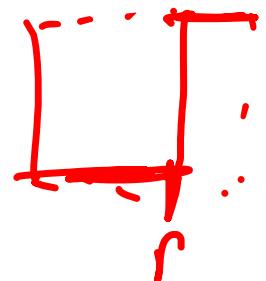
$$\min_{x,y} (E_{x,y}(r,c)) > \Gamma$$

Moreavec (80) - Corner Detection

At corners values of $E_{x,y}$ “are always big”, even for the less significant displacements (x, y)

$$HM(r, c) = T_\gamma \left(\min_{(x,y)} (E_{x,y}(r, c)) \right)$$

Hard Thresholding



where T_γ is the hard thresholding operator having threshold γ

Corner (keypoint) Detection: Look for local maxima of $HM(r, c)$, as corners maximizes this measure



Moravec Drawbacks – Solutions

The response may be **noisy**

$$E_{x,y}(r,c) = \sum_{(u,v) \in U_{r,c}} w_{r,c}(u,v) [I(u,v) - I(u-x, v-y)]^2$$

Solution: take $w_{r,c}$ as Gaussian distributed weights.

Moravec Drawbacks – Solutions

The response is anisotropic since only a finite set of displacements (x, y) is considered

$$E_{x,y}(r,c) = \sum_{(u,v) \in U_{r,c}} w_{r,c}(u,v) [I(u,v) - I(u-x, v-y)]^2$$

therefore, the same corner rotated may yield different responses.

Solution: Expand $I(u - x, v - y)$ in Taylor series

$$I(u - x, v - y) = I(u, v) + xI_x(u, v) + yI_y(u, v) + O(x^2, y^2)$$

where $I_x(\cdot) = \frac{\partial}{\partial x} I(\cdot)$ and $I_y(\cdot) = \frac{\partial}{\partial y} I(\cdot)$, then

$$E_{x,y}(r,c) = \sum_{u,v \in U_{r,c}} w_{r,c}(u,v) \left(xI_x(u, v) + yI_y(u, v) + O(x^2, y^2) \right)^2$$

Moravec Drawbacks – Solutions

We consider only the first-order terms in the Taylor expansion

$$E_{x,y}(r, c) \underset{\text{red}}{\approx} \sum_{u,v \in U_{r,c}} w_{r,c}(u, v) \left(\underbrace{x I_x(u, v)}_{\text{red}} + \underbrace{y I_y(u, v)}_{\text{red}} \right)^2$$

Basic calculus leads to $E_{x,y}(r, c)$

$$\begin{aligned} & \approx \sum_{(u,v) \in U_{r,c}} w_{r,c}(u, v) \left(\underbrace{x^2 I_x^2(u, v)}_{(\text{I } \star \omega)(r, \cdot)} + \underbrace{y^2 I_y^2(u, v)}_{(\text{I } \star \omega)(\cdot, r)} + \boxed{2xy I_x(u, v) I_y(u, v)} \right) \\ & \approx x^2 \boxed{\sum_{(u,v) \in U_{r,c}} w_{r,c}(u, v) I_x^2(u, v)} + y^2 \sum_{(u,v) \in U_{r,c}} w_{r,c}(u, v) I_y^2(u, v) + \\ & \quad + 2xy \sum_{(u,v) \in U_{r,c}} w_{r,c}(u, v) I_x(u, v) I_y(u, v) \end{aligned}$$

Moravec Drawbacks – Solutions

Which is an expression that admits the following matrix notation

$$E_{x,y}(r, c) \approx [x, y] M_{r,c} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$(r, c) \begin{bmatrix} \cdot \\ \cdot \end{bmatrix} M_{r,c}$$

where

$$\begin{aligned} M_{r,c} &= \begin{bmatrix} (I_x^2 \odot w)(r, c) & (I_x I_y \odot w)(r, c) \\ (I_x I_y \odot w)(r, c) & (I_y^2 \odot w)(r, c) \end{bmatrix} \\ &\doteq \begin{bmatrix} I_x^2 \odot w & I_x I_y \odot w \\ I_x I_y \odot w & I_y^2 \odot w \end{bmatrix} (r, c) \end{aligned}$$

$$\begin{array}{c} I_x^2 \\ I \\ \swarrow \quad \searrow \\ I_x I_y \odot w \\ \downarrow \\ I_x \odot I_y \end{array}$$

Note that I_x and I_y denotes image derivatives, which can be computed with any derivative filters (Sobel, Previtt, Gaussian)

$[x, y]$ always denotes the displacement (very bad notation, sorry)

Moravec Drawbacks – Solutions

Thus $E_{x,y}(r, c)$ can be computed at any pixel (r, c) , w.r.t. any displacement vector (x, y)

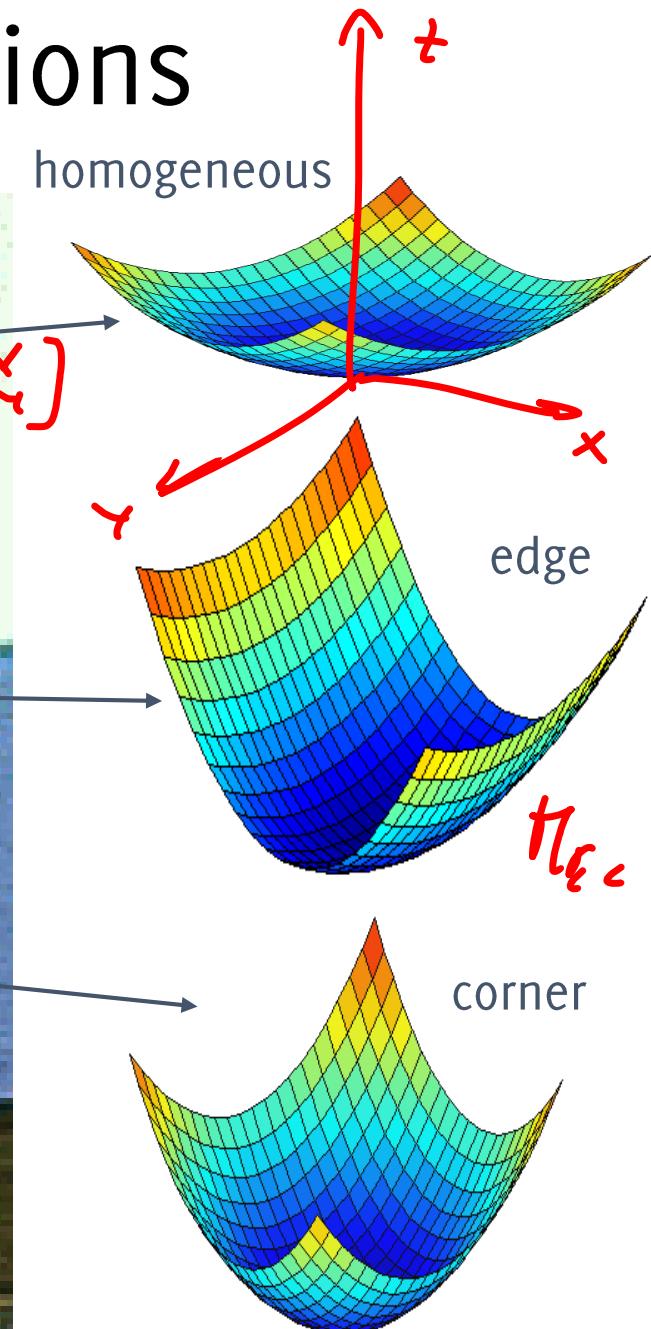
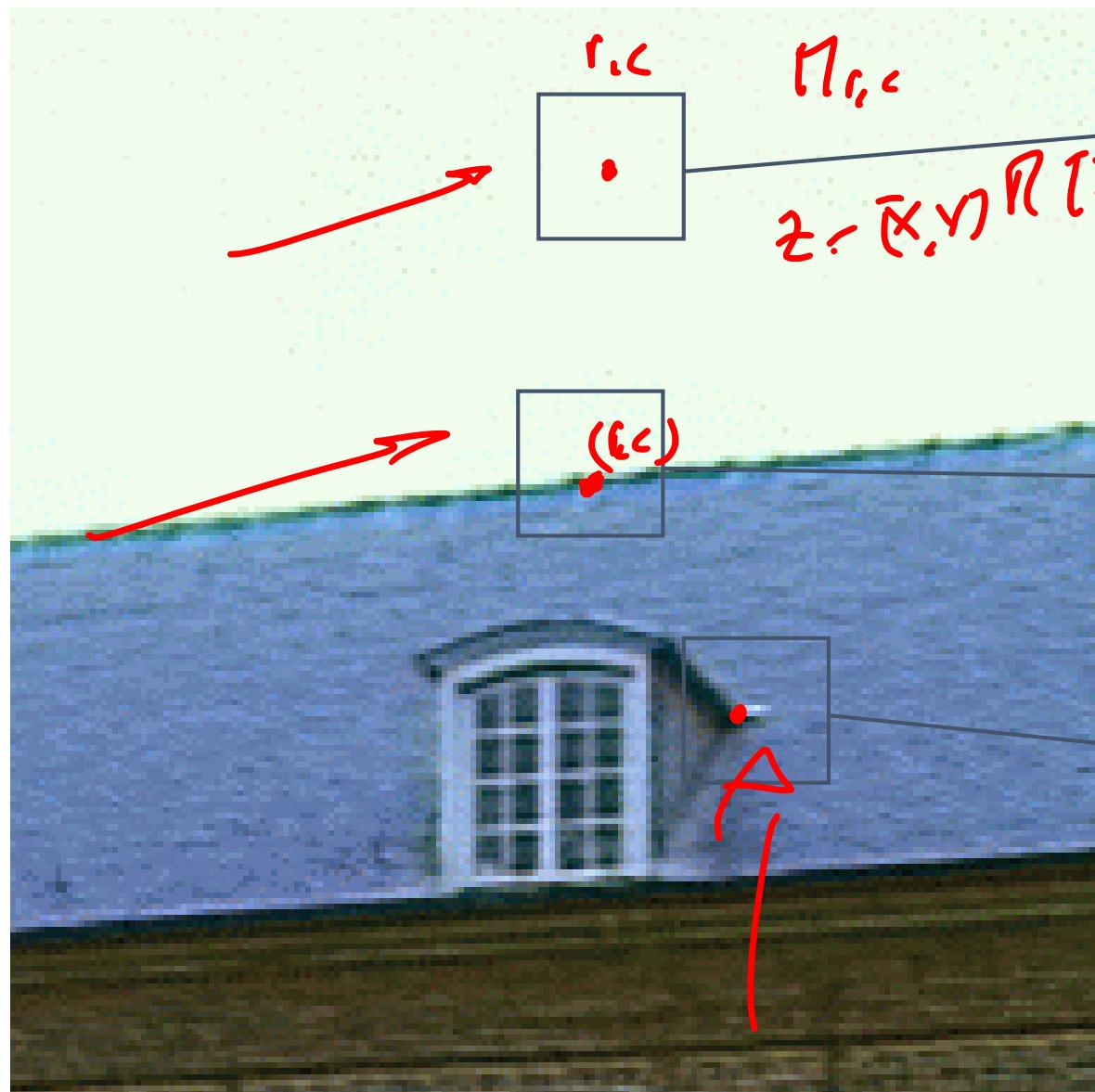
$$E_{x,y}(r, c) \approx [x, y] \begin{bmatrix} (I_x^2 \odot w)(r, c) & (I_x I_y \odot w)(r, c) \\ (I_x I_y \odot w)(r, c) & (I_y^2 \odot w)(r, c) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

The response $E_{x,y}(r, c)$ w.r.t. any displacement (x, y) can be approximated by the quadratic expression involving the matrix $M_{r,c}$ in any pixel (r, c) .

Obtaining the matrix $M_{r,c}$ is straightforward, as it involves only computing (few) image derivatives.

Matrix M values in different image regions

The “*analytical behavior*” of the matrix $M_{r,c}$ in different locations r, c



Corner Detection Goal

Corner Detection: Find points for which the following is maximum

$$\min_{[x,y]} [x, y] M \begin{bmatrix} x \\ y \end{bmatrix}, \text{ given } \| [x, y] \|_2 = 1$$

$\lambda_1 < \lambda_2$ i.e. maximize smallest eigenvalue of M
 $\lambda_1 > \lambda_2$ good $\propto \lambda_1 \gg \lambda_2$

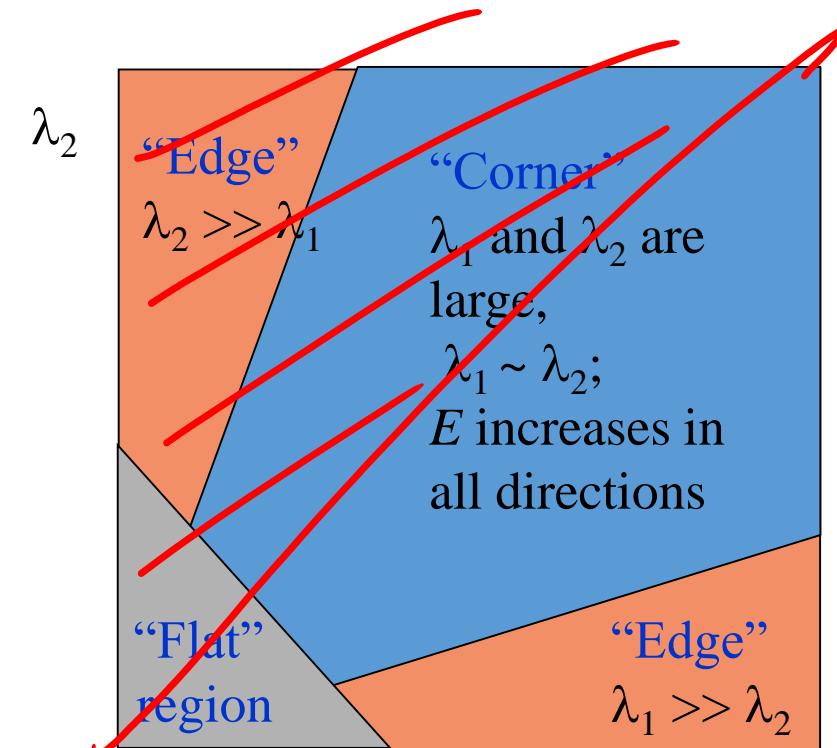
$$\Sigma = U \Sigma V \quad \Sigma = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

Moravec Drawbacks – Solutions

Considering the minimum of E is not a great deal, may give too **ready responses**, and might require many calculations, since many displacements (x, y) have to be considered.

Solution:

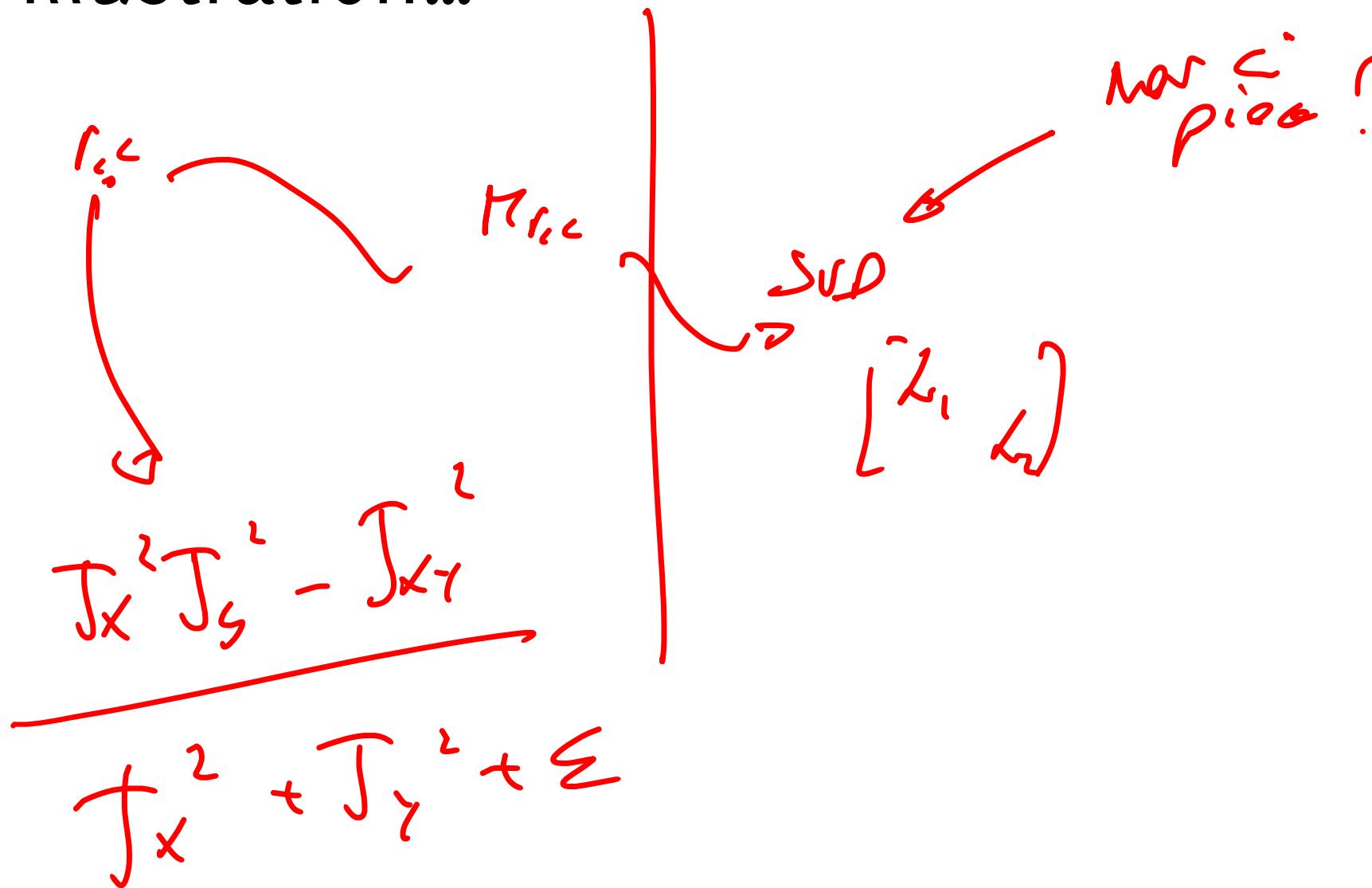
- consider the $SVD(M_{r,c})$ and require that the minimum eigenvalue of $M_{r,c}$ is **large at corners**
- This means that $E_{x,y}(r, c)$ exhibits a large variation w.r.t. any displacement vector (x, y)



Being λ_1 and λ_2 the eigenvalues of M

λ_1

An Illustration...



Harris – Stevens (88)

The following relation holds

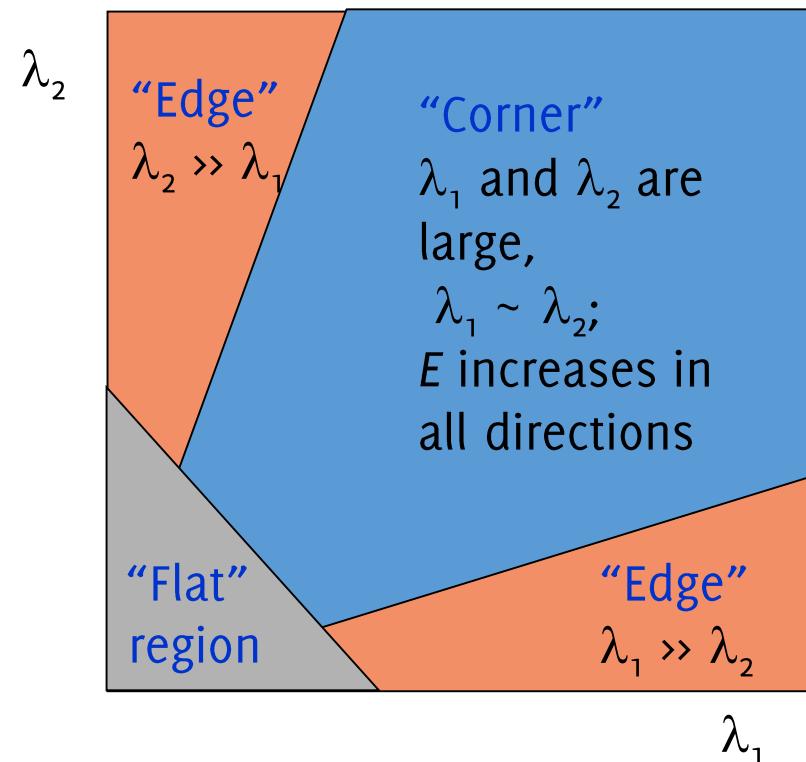
$$Tr(M) = \lambda_1 + \lambda_2$$

$$\det(M) = \lambda_1 \cdot \lambda_2$$

And the function

$$\boxed{\det(M) - k Tr(M)}$$

is large when both λ_1 and λ_2 are large, where
 $k = 0.04$.



Our Matrix

Thus $E_{x,y}(r, c)$ can be computed at any pixel (r, c) , w.r.t. any displacement vector (x, y) by using the following matrix

$$M_{r,c} = \begin{bmatrix} (I_x^2 \odot w)(r, c) & (I_x I_y \odot w)(r, c) \\ (I_x I_y \odot w)(r, c) & (I_y^2 \odot w)(r, c) \end{bmatrix}$$

if we define,

$$J_x^2 = I_x^2 \odot w, \quad J_y^2 = I_y^2 \odot w, \quad J_{xy} = I_x I_y \odot w$$

the following relations hold

$$\text{Tr}(M_{r,c}) = J_x^2(r, c) + J_y^2(r, c) = ((I_x^2 + I_y^2) \odot w)(r, c)$$

$$\det(M_{r,c}) = J_x^2 J_y^2(r, c) - J_{xy}^2(r, c)$$

Harris – Stevens (88)

The following relation holds

$$Tr(M) = \lambda_1 + \lambda_2$$

$$\det(M) = \lambda_1 \cdot \lambda_2$$

And the function

$$\det(M) - k Tr(M)$$

is large when both λ_1 and λ_2 are large, where $k = 0.04$.

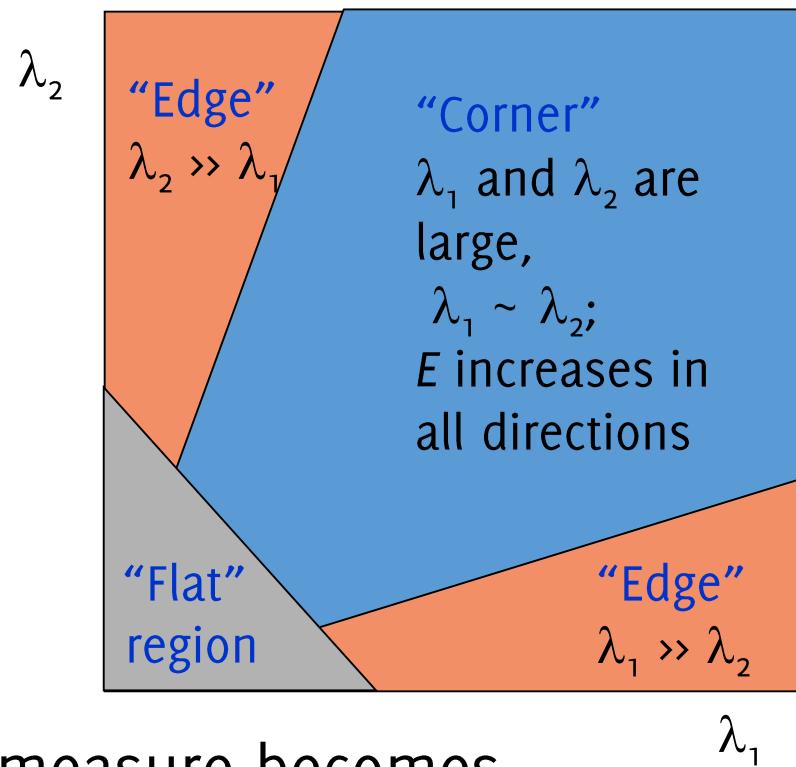
Let $J_x^2 = I_x^2 \odot w$, $J_y^2 = I_y^2 \odot w$

$$J_{xy} = I_x I_y \odot w$$

It is possible to avoid computing $SVD(M)$ and the Harris measure becomes

$$CIM = (J_x^2 J_y^2 - J_{xy}^2) - k (J_x^2 + J_y^2)$$

defined as in the previous slide



Harris – Stevens (88)

The following relation holds \propto

$$Tr(M) = \lambda_1 + \lambda_2$$

$$\det(M) = \lambda_1 \cdot \lambda_2$$

And the function

$$\det(M) - k Tr(M)$$

is large when both λ_1 and λ_2 are large, where $k = 0.04$.

$$\text{Let } J_x^2 = I_x^2 \odot w, J_y^2 = I_y^2 \odot w$$

Gaussian filter

$$J_{xy} = I_x I_y \odot w$$

It is possible to avoid computing $SVD(M)$ and the Harris measure becomes

$$CIM = \boxed{(J_x^2 J_y^2 - J_{xy}^2) - k (J_x^2 + J_y^2)}$$

defined as in the previous slide

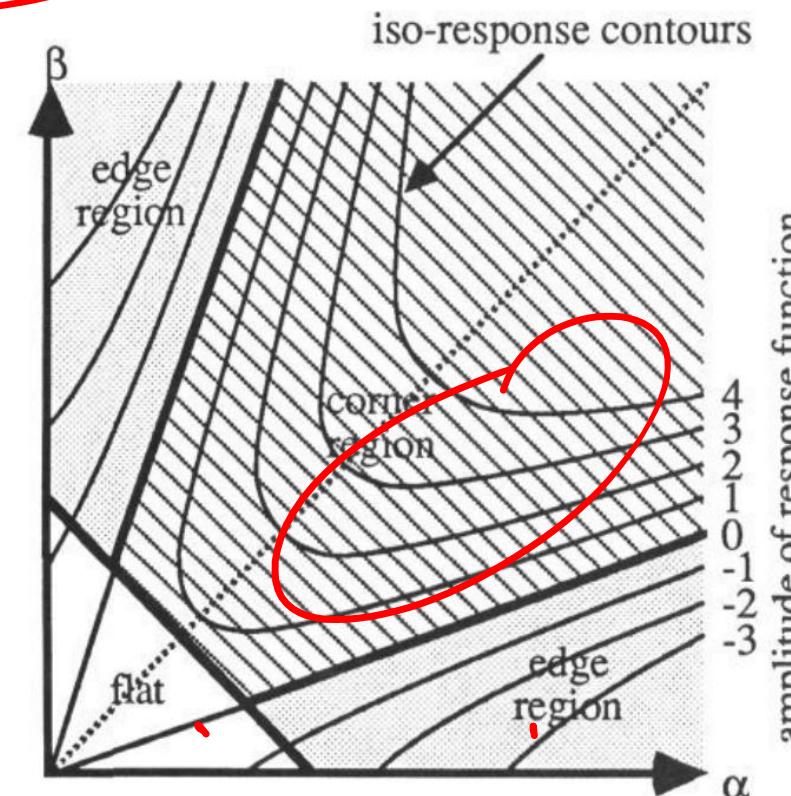


Figure from
Harris '88

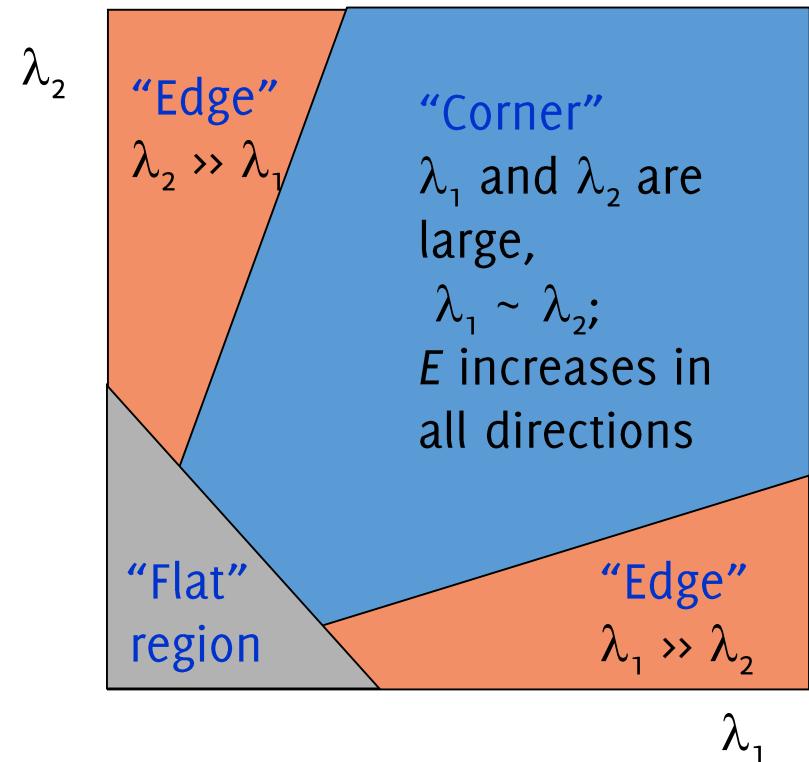
Harris – Stevens (88)

Alternatively, Noble's variant which does not involve k :

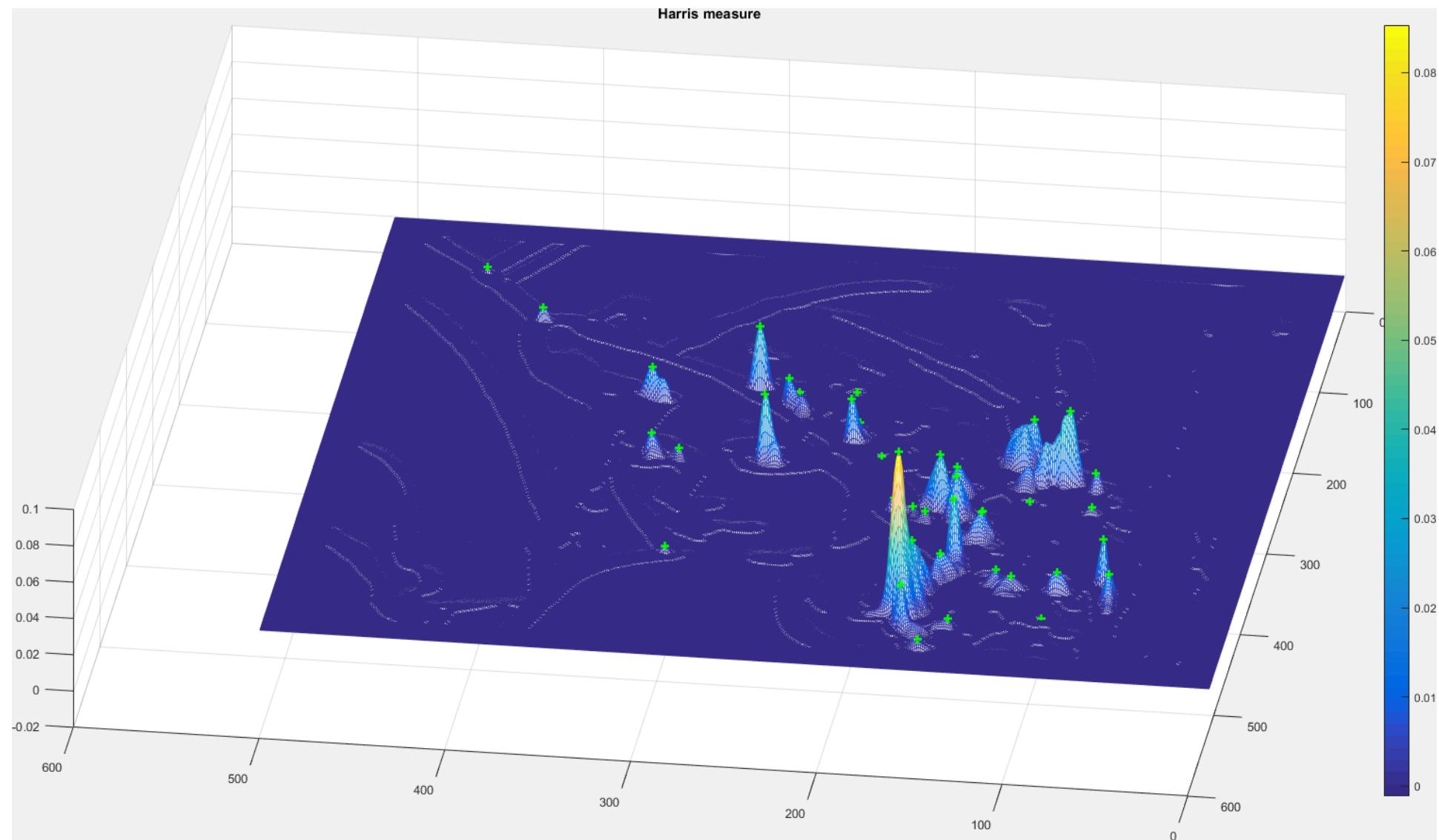
$$CM = \frac{\det(M)}{Tr(M) + \epsilon}$$

That can thus be computed from the image derivatives as:

$$CM = \frac{(J_x^2 J_y^2 - J_{xy}^2)}{J_x^2 + J_y^2 + \epsilon}$$



Extract Local Maxima of Harris Corner Measure



Intermezzo: How to find local maxima?

Local Maxima of Harris Corner Measure



Scale-Invariant Feature Transform

Giacomo Boracchi

CVPR USI, April 21 2020

Histograms of Oriented Gradients (HOG)

HOG: a Family of **Image Features** that are built upon orientation of image gradients around selected keypoints

SIFT [Lowe 2004] is a prominent example of HOG features. SIFT features are invariant to:

- image scale
- Image rotation

The cost of extracting SIFT is minimized by a **cascade approach**, in which the more expensive operations are applied only at locations that pass an initial test.

Scale Invariant Feature Transform

SIFT

SIFT that are shown to provide **robust matching** across a

- substantial range of affine distortions,
- change in 3D viewpoint,
- addition of noise,
- change in illumination.

The **SIFT descriptors** are highly distinctive, relatively easy to extract and allow for correct object identification with low probability of mismatch.

Scale invariance is provided by an ad-hoc keypoint extraction algorithm

SIFT Outline

SIFT generates large numbers of features that densely cover the image over the full range of scales and locations.

It is composed of the following steps

• Scale-space extrema detection

• Keypoint localization

• Orientation assignment

• Keypoint descriptor

≈ Keypoint detection

Scale-space extrema detection

SIFT Scale Invariant Feature Transform [Lowe 2004]

SIFT Outline

Scale-space extrema detection: search over all the scales and image locations for potential interest points that are invariant to scale and orientation.

Keypoint localization: At each candidate location, a detailed model is fit to determine location and scale

Orientation assignment: One or more orientations are assigned to each keypoint location based on local image gradient directions.

Keypoint descriptor: The local image gradients are measured at the selected scale in the region around each keypoint

SIFT generates large numbers of features that densely cover the image over the full range of scales and locations

Detection of scale-space extrema

Keypoint detection is the first stage of a **cascade approach**

The goal is to identify locations and scales that can be repeatedly assigned under differing views of the same object.

How: search for **stable keypoints** across all possible scales of the image, i.e., in the **scale space**



Image Pyramid

Unfortunately, only a single image from a single scale is available. How to extract information from “all possible scales”?

By **generating an image pyramid**: Build different representations of the original image at different resolutions/zoom levels, by convolution

- The highest resolution corresponds to the original image I
- Lower resolutions are synthetically generated through blurring by convolution and resampling

An image pyramid is obtained by convolving the image I with several Gaussian kernels G_σ having standard deviation σ .

We define the layers of such pyramid as:

$$L(x, y, \sigma) = (G_\sigma \circledast I)(x, y)$$

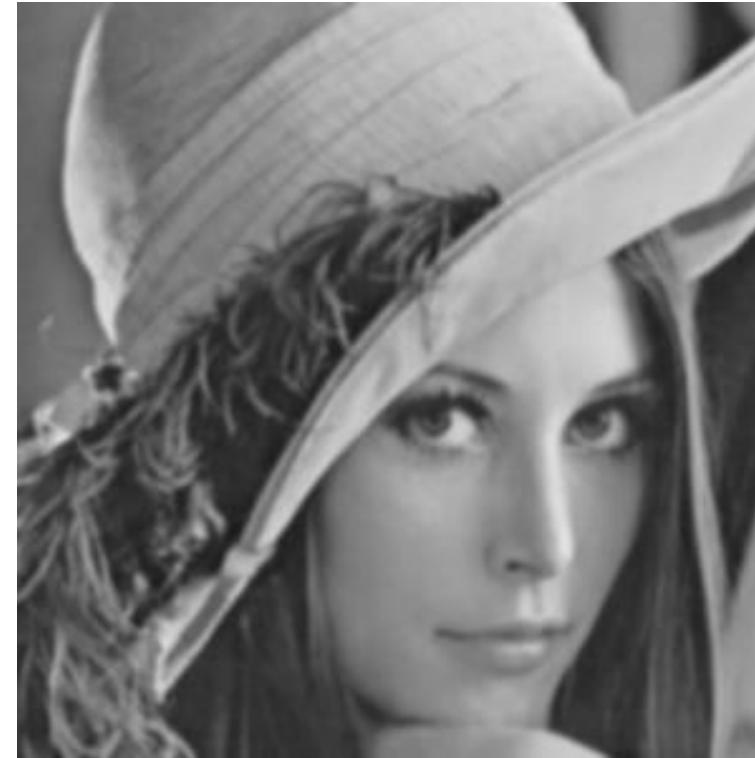
An Image Pyramid

$L(\cdot, \cdot, 1)$



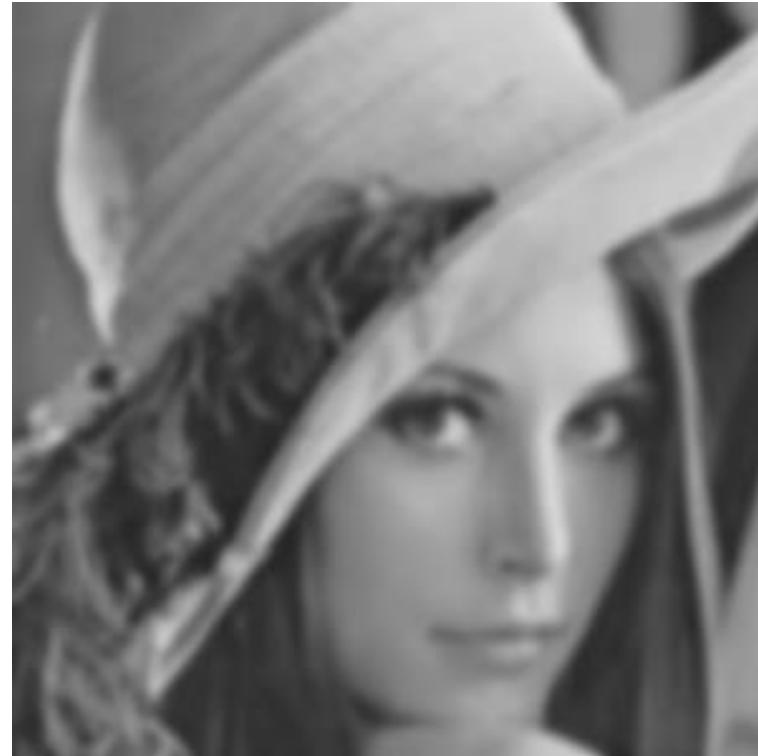
An Image Pyramid

$L(\cdot, \cdot, 2)$



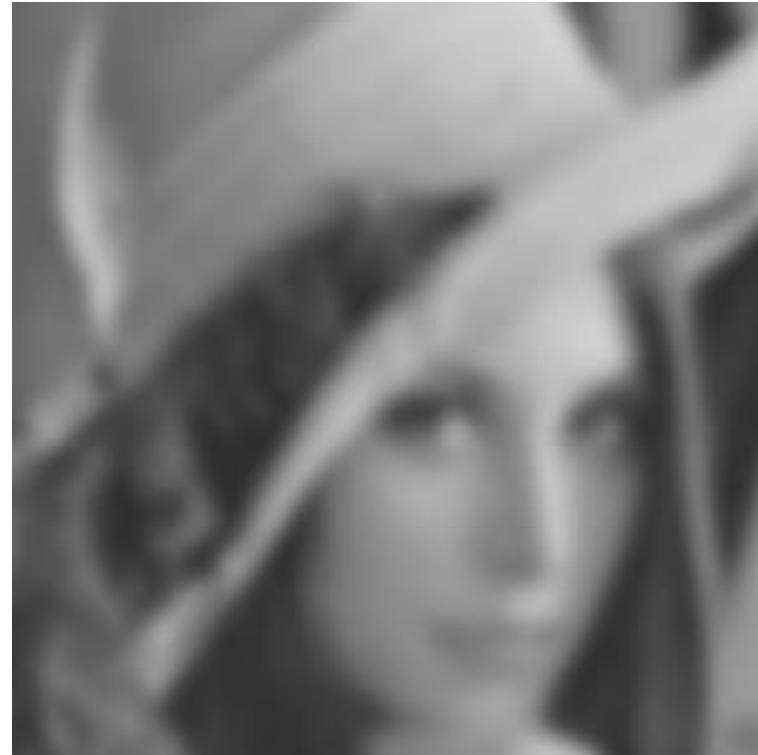
An Image Pyramid

$L(\cdot, \cdot, 3)$



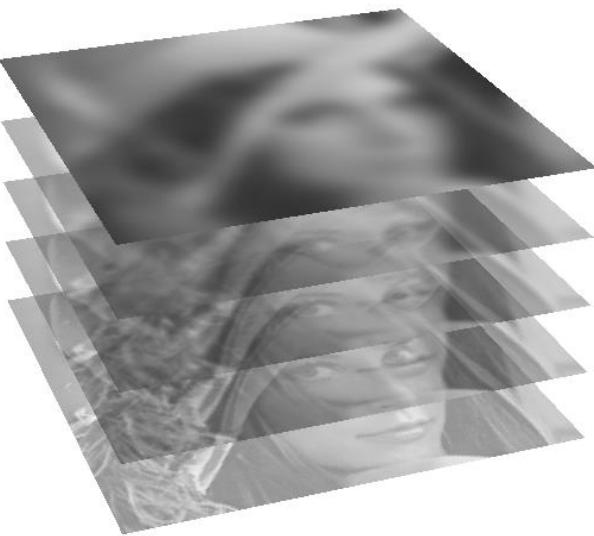
An Image Pyramid

$L(\cdot, \cdot, 4)$



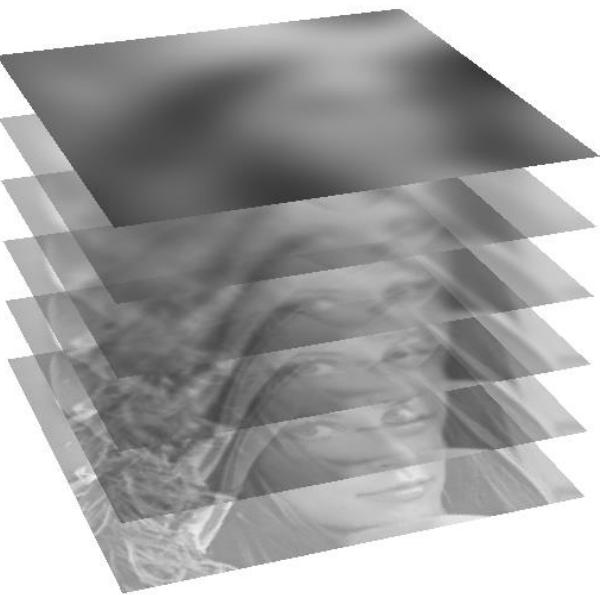
An Image Pyramid

$L(\cdot, \cdot, 5)$



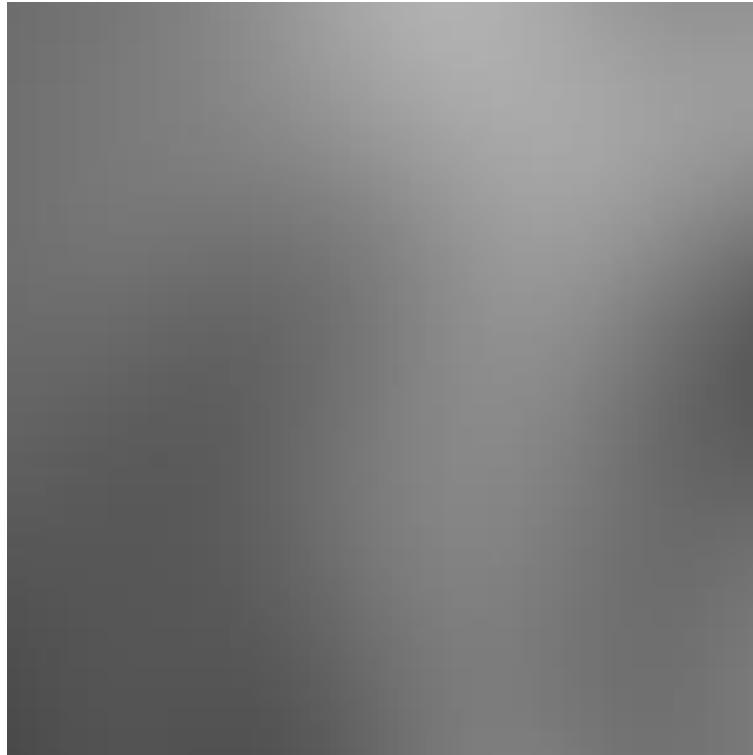
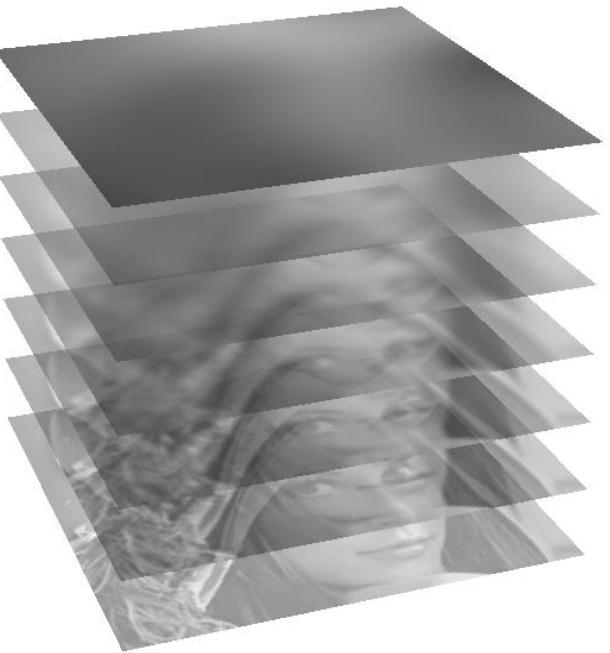
An Image Pyramid

$L(\cdot, \cdot, 6)$

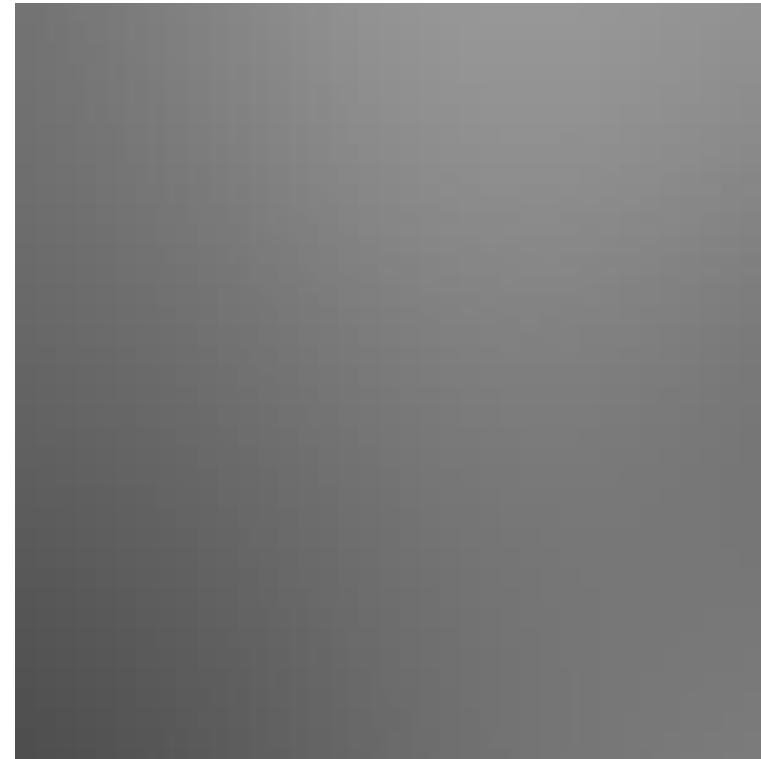
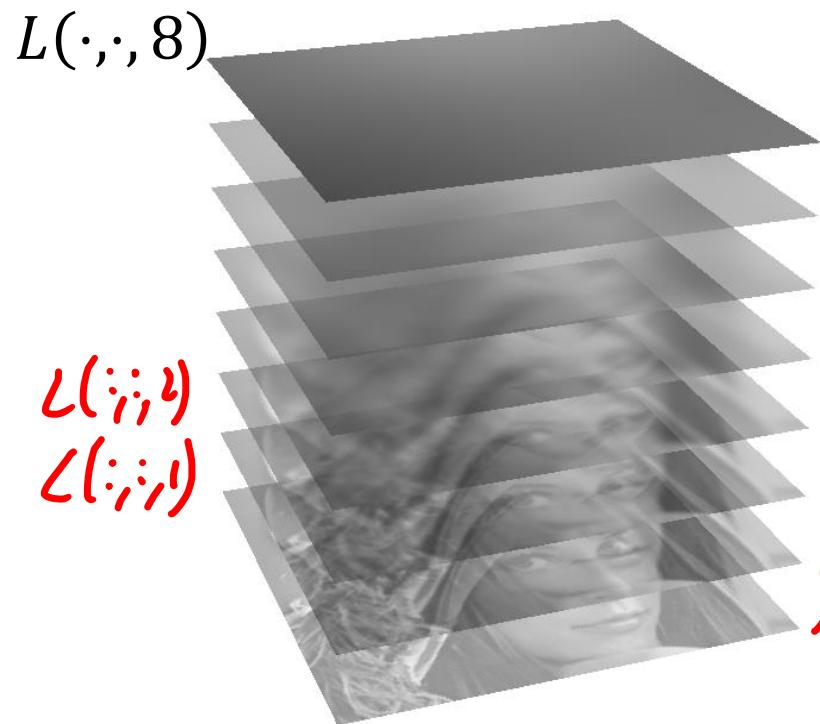


An Image Pyramid

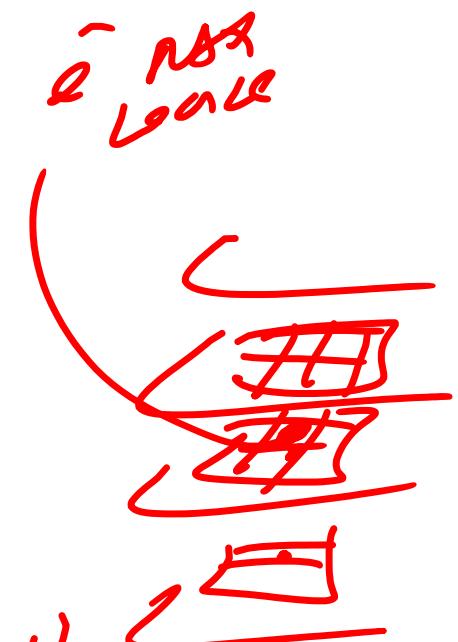
$L(\cdot, \cdot, 7)$



An Image Pyramid



$$D(\cdot, \cdot, r) = L(\cdot, \cdot, r) - L(\cdot, \cdot, 1)$$



Keypoint Localization in the Scale Space

Keypoints are detected as the local maxima in the difference between two adjacent representations in the scale space

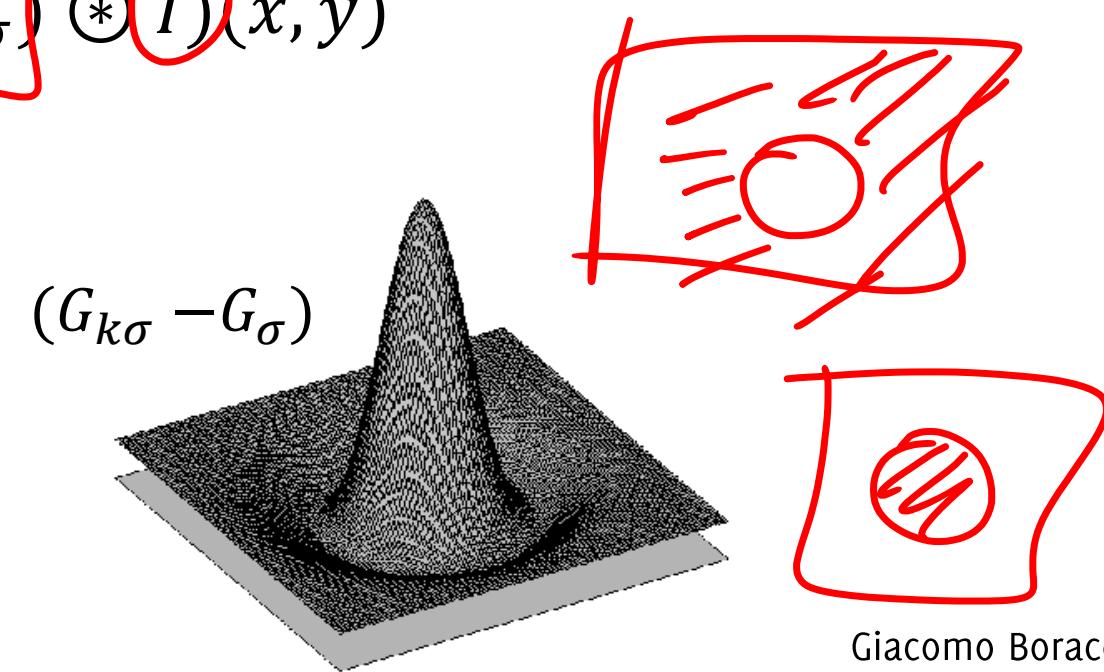
$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma) \quad (G_{k\sigma} * I) - (G_\sigma * I)$$

That, thanks to convolution properties we have that:

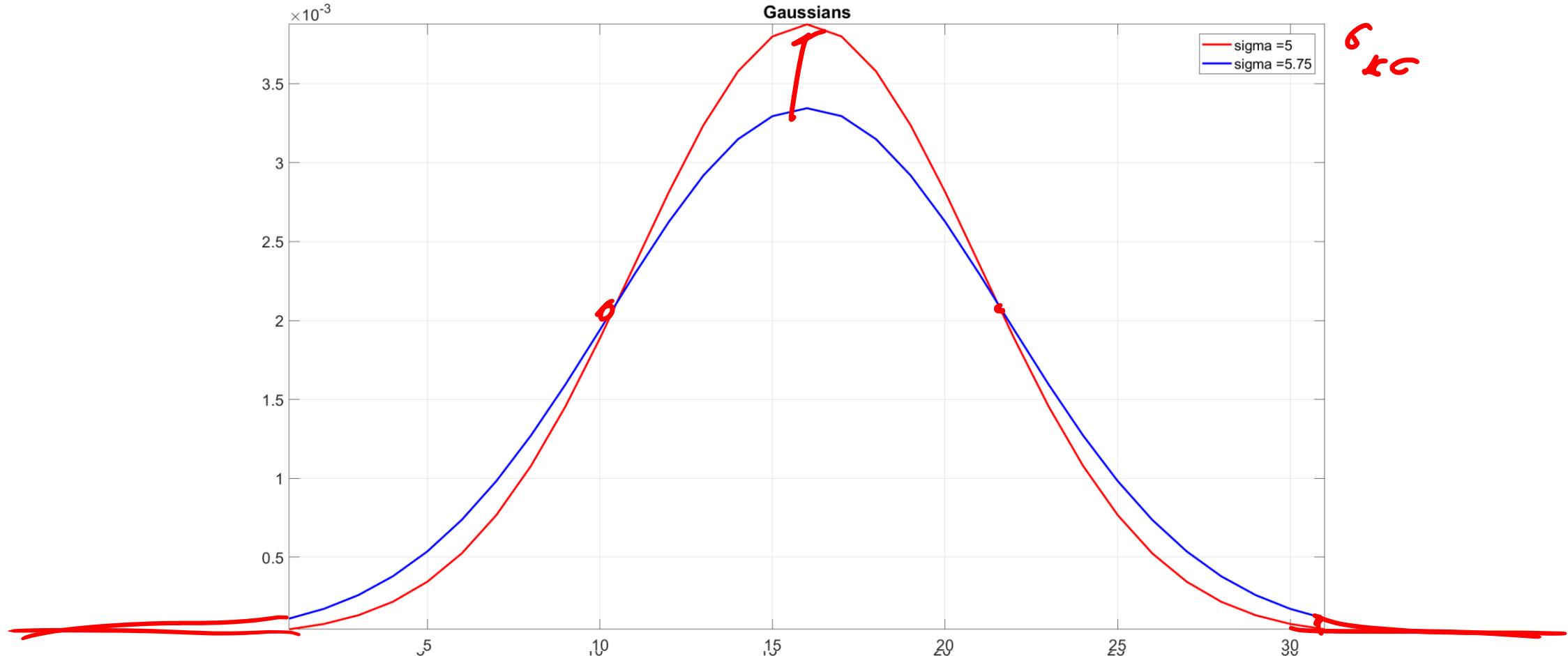
$$D(x, y, \sigma) = ((G_{k\sigma} - G_\sigma) \odot I)(x, y)$$

What about $(G_{k\sigma} - G_\sigma)$?

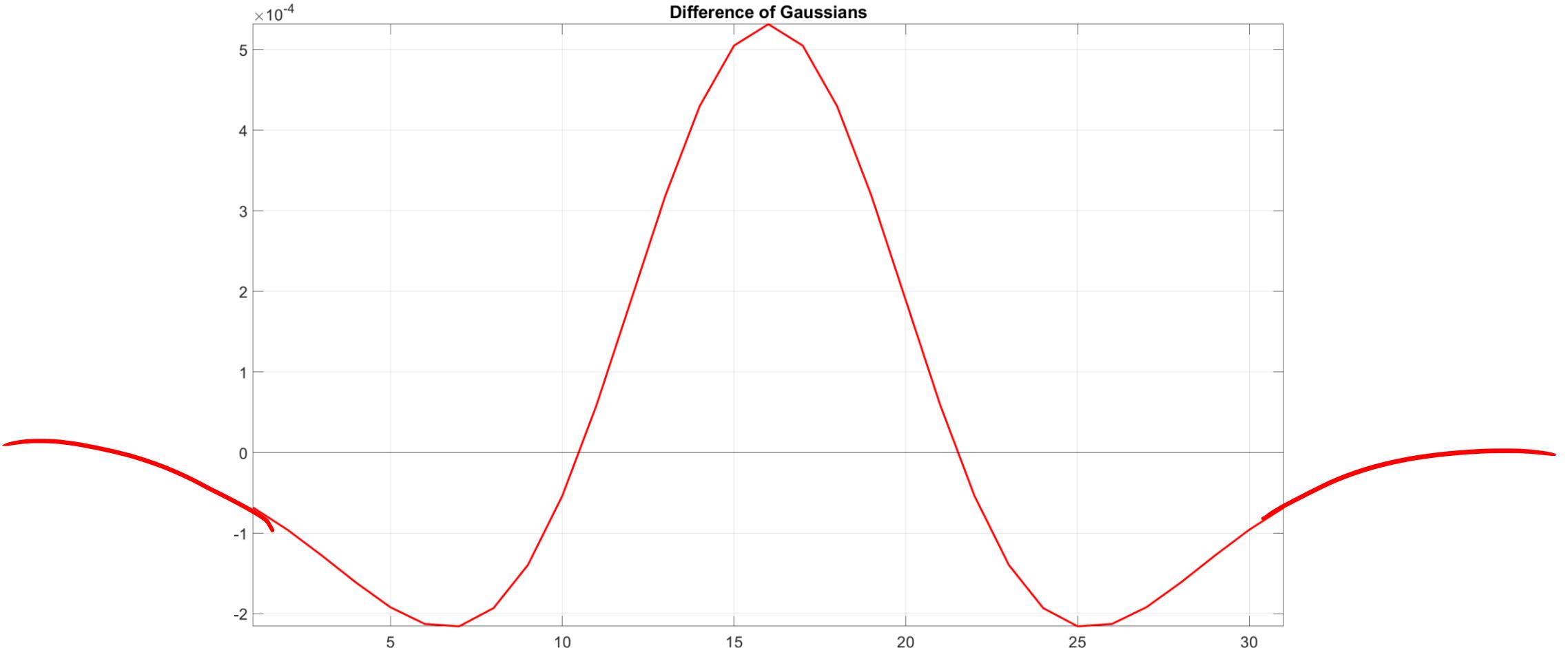
It is the filter corresponding to a
difference-of-Gaussians:
it acts as a “blob” detector



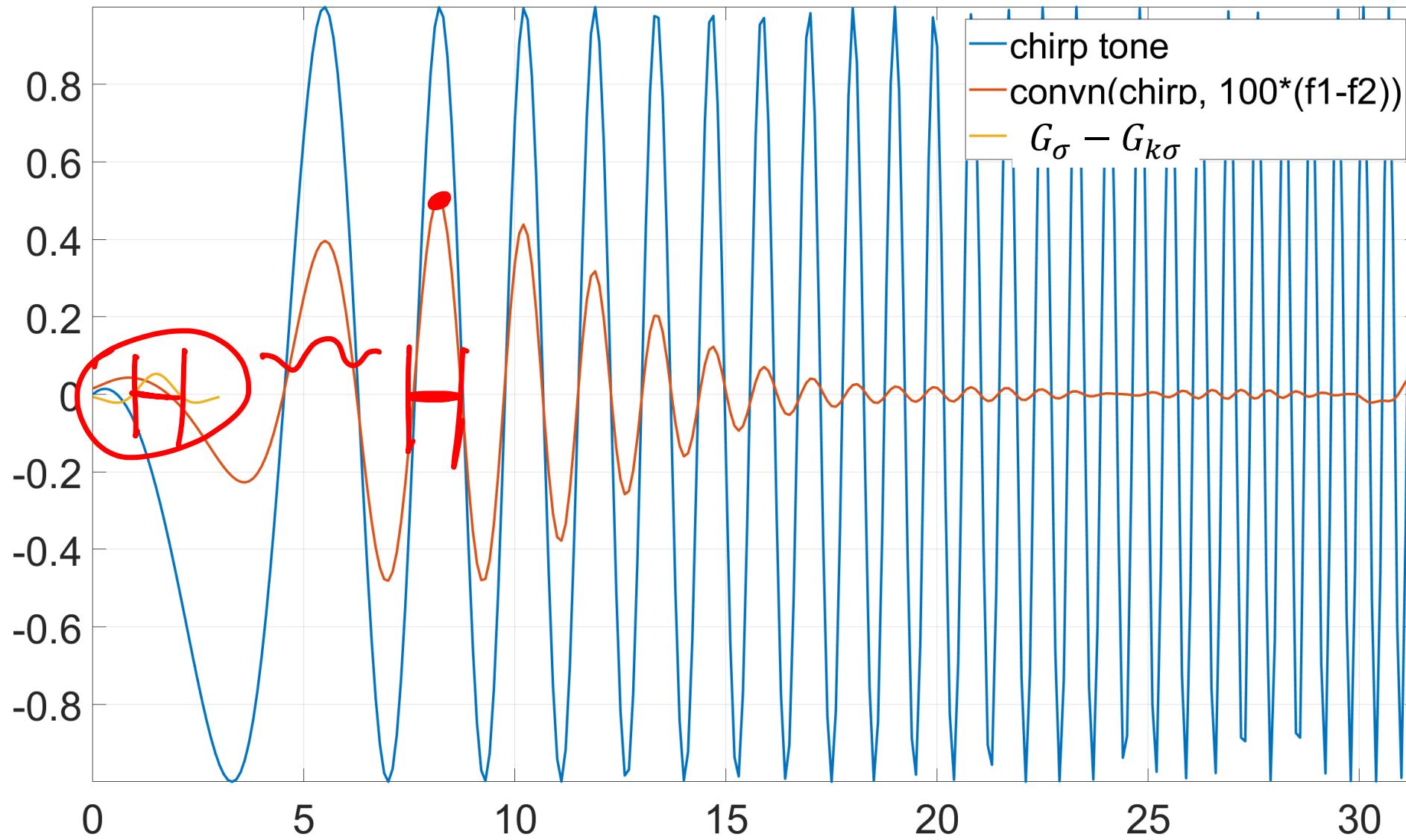
Let's look at a 1d-example



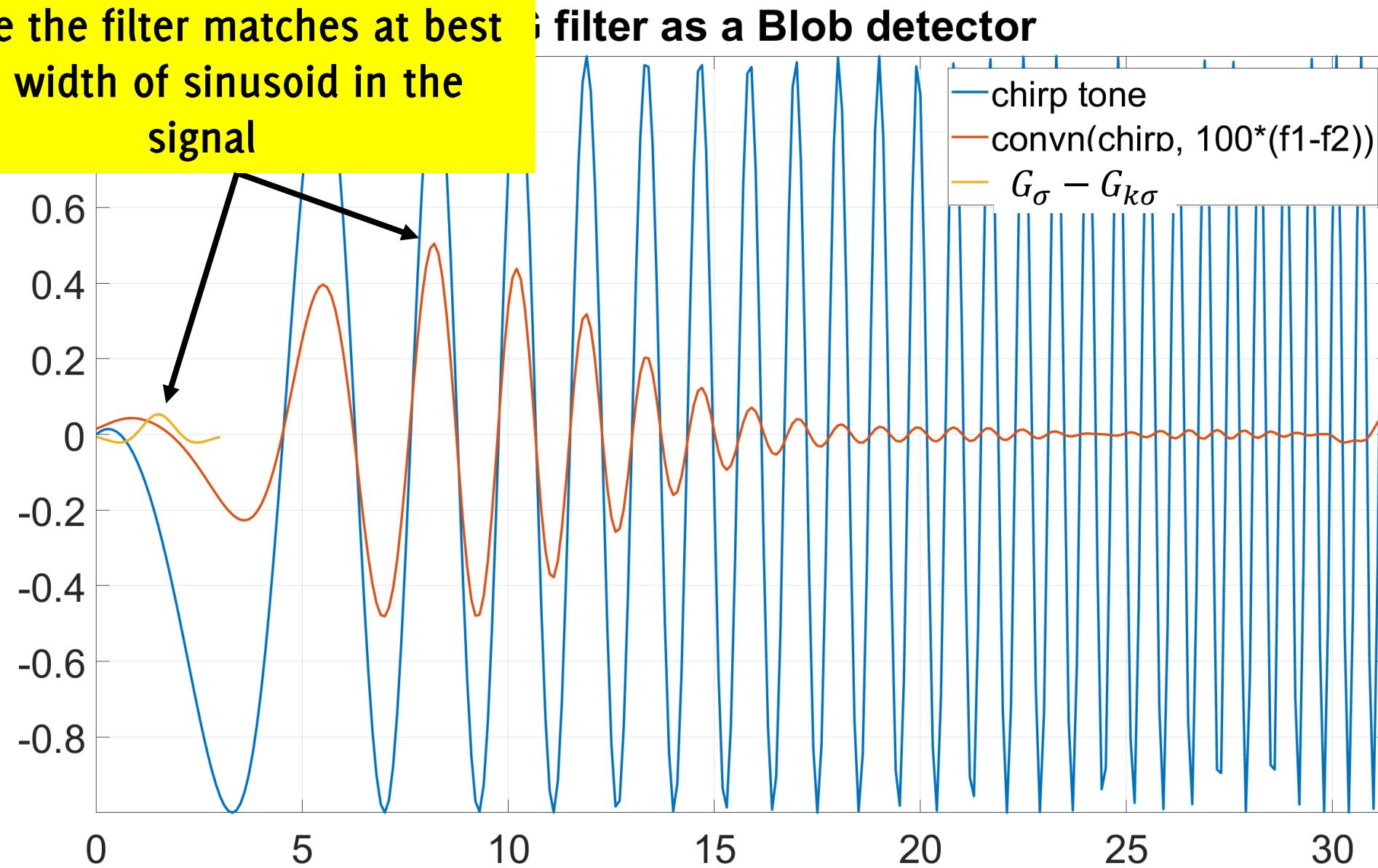
Let's look at a 1d-example



DoG filter as a Blob detector

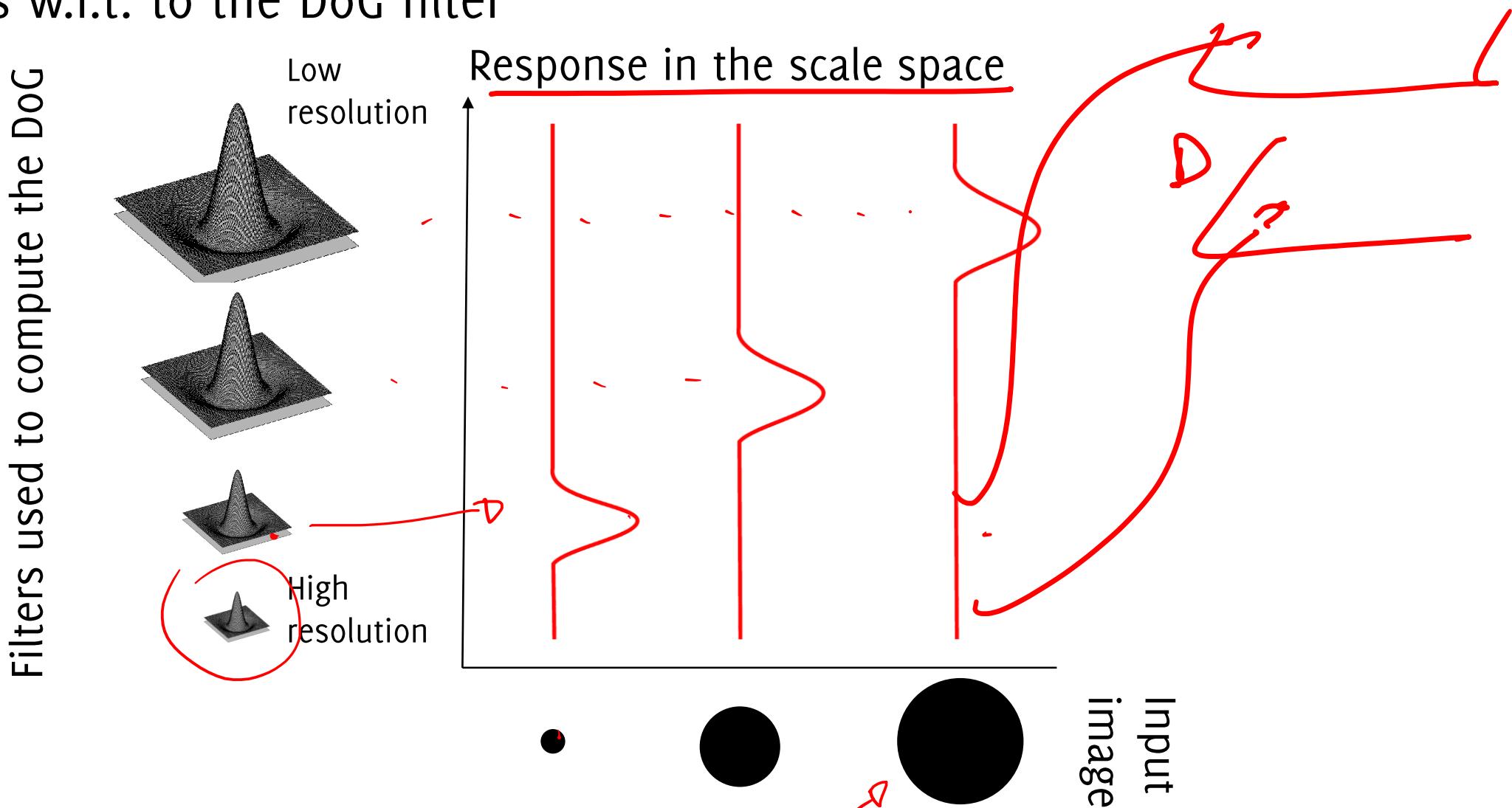


Convolution achieve its maximum where the filter matches at best the width of sinusoid in the signal

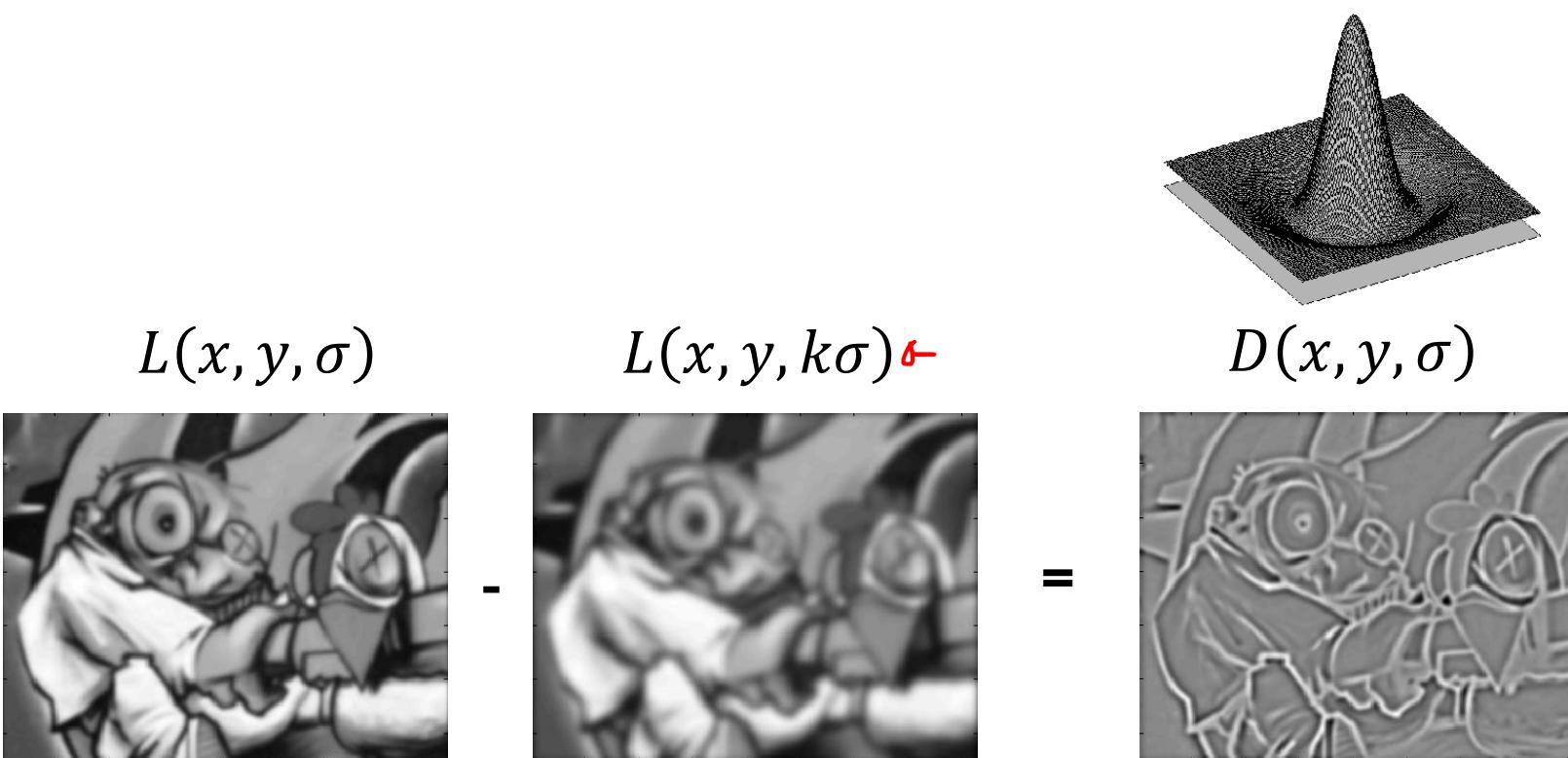


Difference-of-Gaussian

Responses w.r.t. to the DoG filter

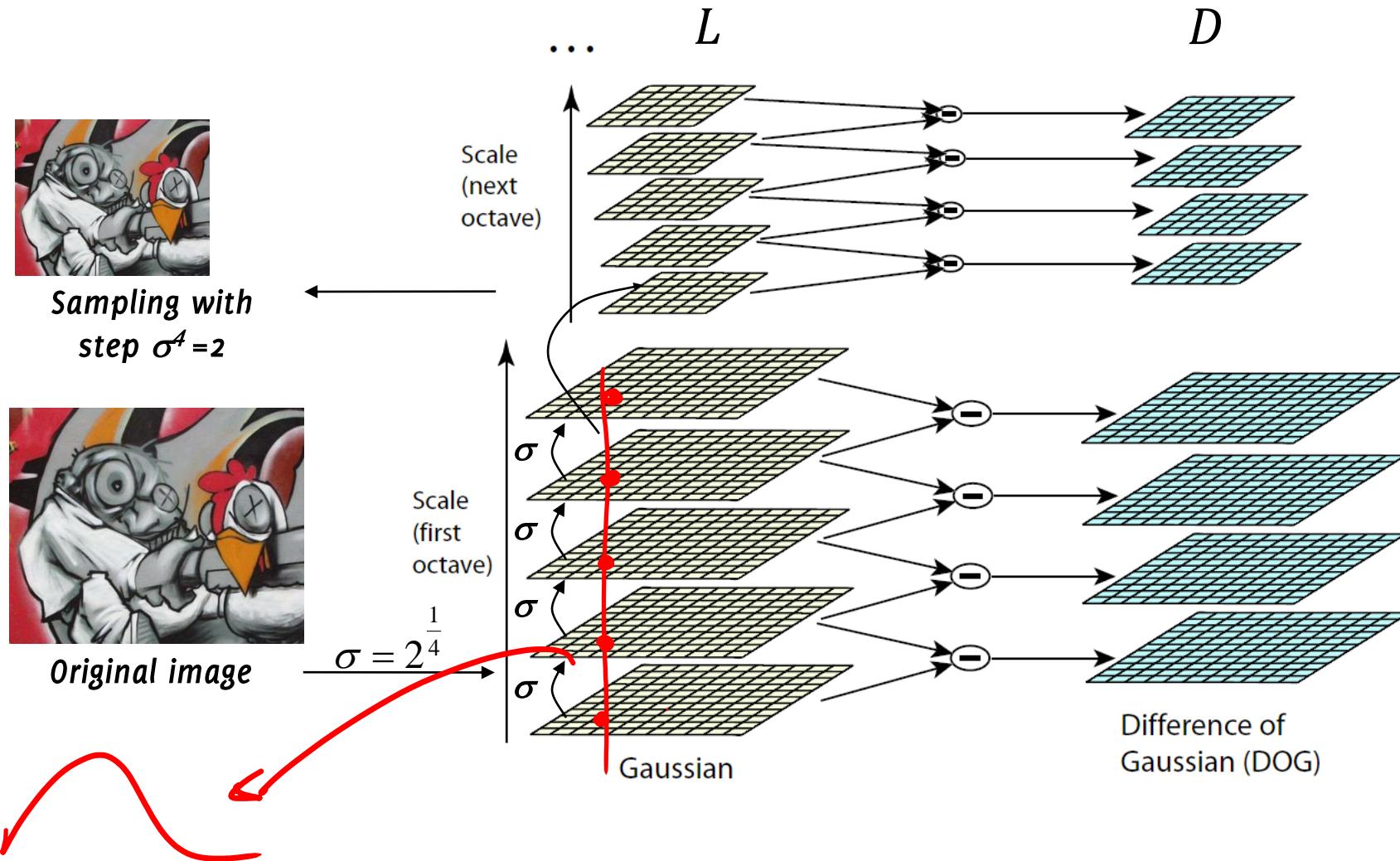


Difference-of-Gaussian (DoG)



DoG - Efficient Computation

Computation in Gaussian scale pyramid



Advantages of the Difference of Gaussian

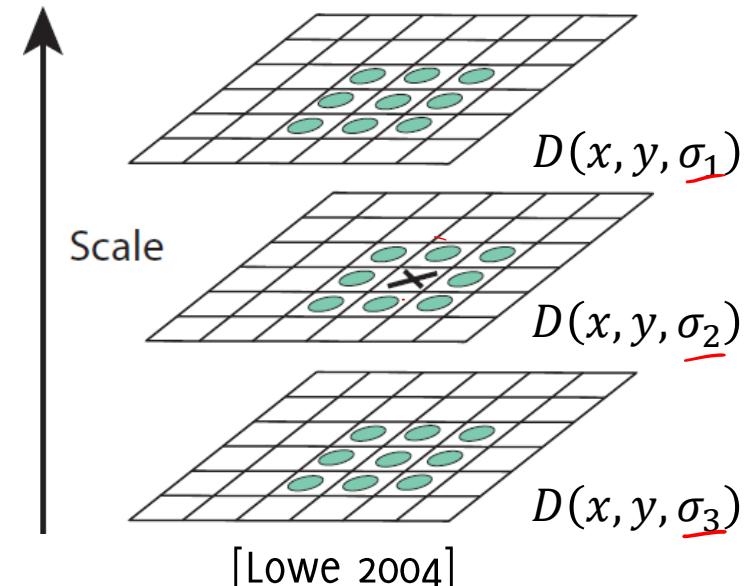
Why the Difference of Gaussian?

- **It is very efficient** to compute since
 - Smoothed images in the pyramid can be computed in cascade, using smaller filters
 - Smoothed images are also used to define the descriptors
- The **DoG approximates the scale-normalized Laplacian of Gaussian** [see Lowe 2004], whose local maxima and minima have been shown (experimentally) to provide the most stable image features.

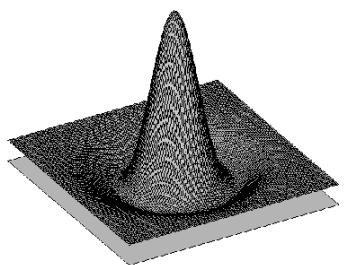
Local Extrema Detection

Local maxima and minima are found by comparing the values of adjacent DoG of the scale space

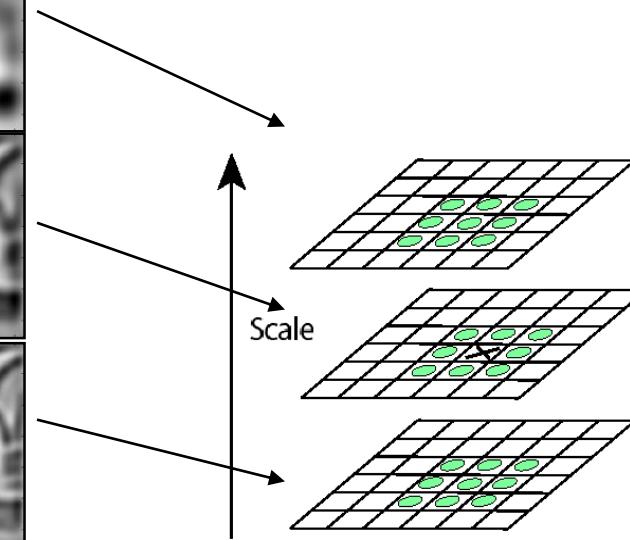
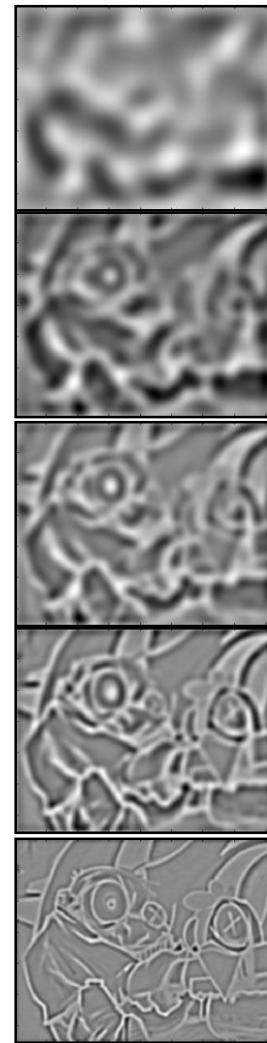
- Each point is compared to its 8 neighbors in the current DoG and 9 neighbors in the scale above and below
- It is selected only if it is larger/smaller than all of these



Local maxima in position-scale space of DoG



$$L_{xx}(\sigma) + L_{yy}(\sigma) \rightarrow \sigma^5$$
$$L_{xx}(\sigma) + L_{yy}(\sigma) \rightarrow \sigma^4$$
$$L_{xx}(\sigma) + L_{yy}(\sigma) \rightarrow \sigma^3$$
$$L_{xx}(\sigma) + L_{yy}(\sigma) \rightarrow \sigma^2$$
$$L_{xx}(\sigma) + L_{yy}(\sigma) \rightarrow \sigma$$



⇒ List of keypoints
 (r, c, s)

Keypoint localization

SIFT Scale Invariant Feature Transform [Lowe 2004]

SIFT outline

Scale-space extrema detection: search over all the scales and image locations for potential interest points that are invariant to scale and orientation.

{(x,y,s)}

Keypoint localization: At each candidate location, a detailed model is fit to determine location and scale

Orientation assignment: One or more orientations are assigned to each keypoint location based on local image gradient directions.

Keypoint descriptor: The local image gradients are measured at the selected scale in the region around each keypoint

SIFT generates large numbers of features that densely cover the image over the full range of scales and locations

The Issue

It is necessary to analyze the nearby data of each candidate keypoint to estimate its:

- location,
- scale,
- ratio of principal curvatures of the image

These information are associated to each keypoint and are used for:

- building the descriptor
- rejecting many keypoints that have low contrast or are poorly localized along an edge.

The issue



$$D(x, y, \sigma) = D(\hat{x}, \hat{y}, \hat{\sigma})$$

To build meaningful feature descriptors, **we need to associate each keypoint to its intrinsic scale** (the layer in the pyramid)

The descriptor refers to a reference scale of each keypoint, this guarantees scale invariance

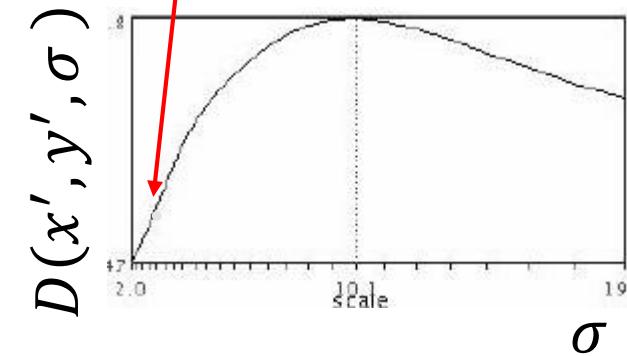
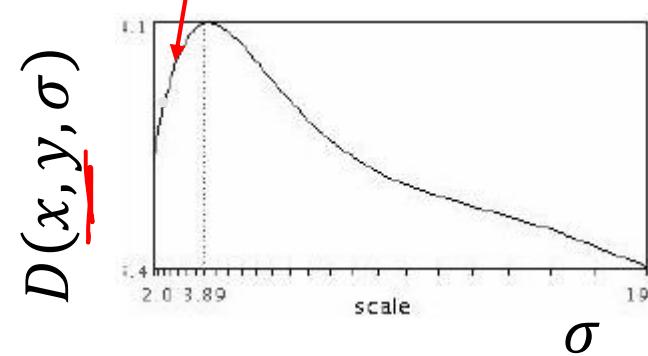
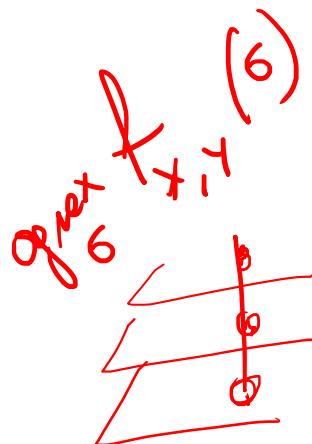
Automatic Scale Selection: Maximizing Function Response

Function responses for increasing scale (scale signature)

I_1

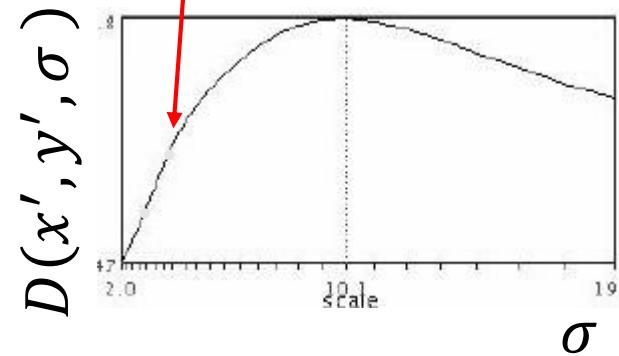
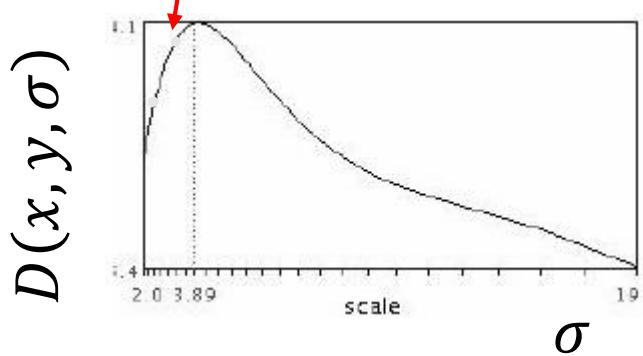
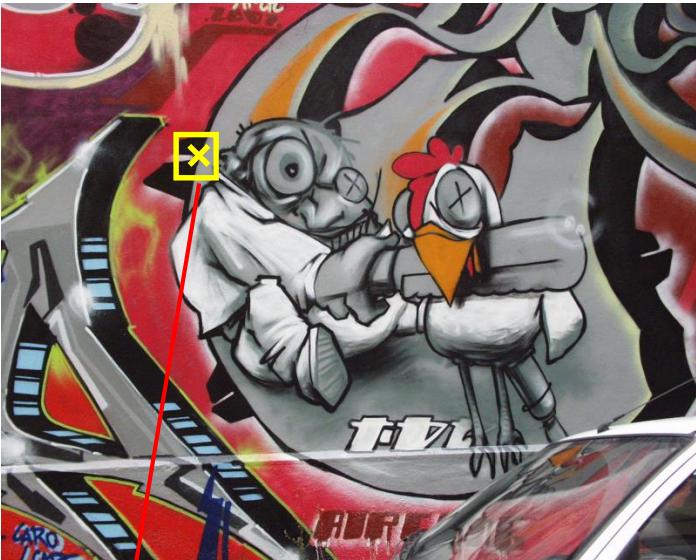


I_2



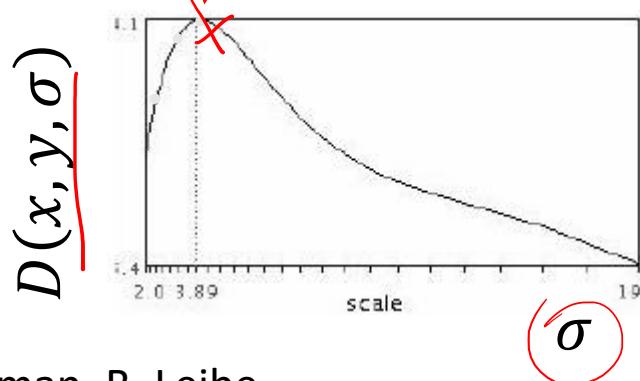
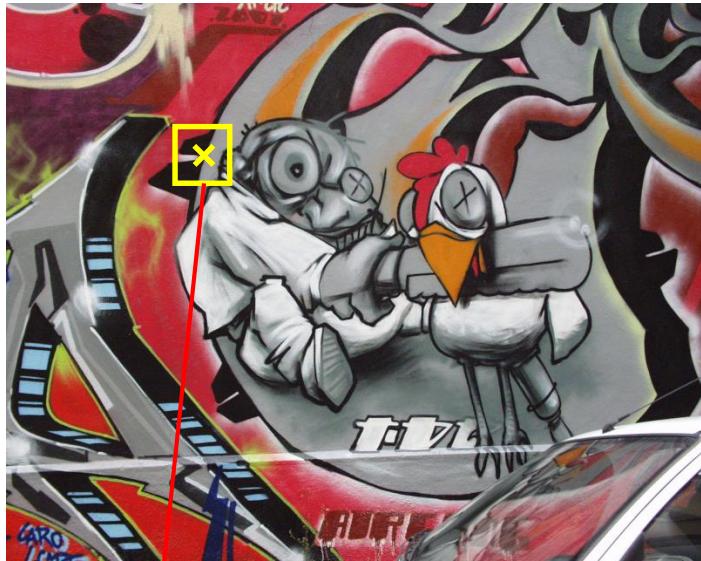
Automatic Scale Selection

Function responses for increasing scale (scale signature)

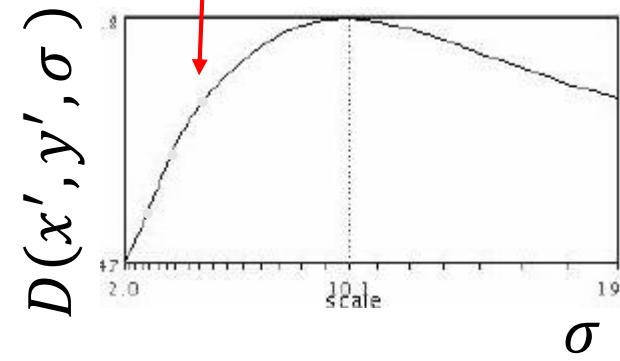


Automatic Scale Selection

Function responses for increasing scale (scale signature)



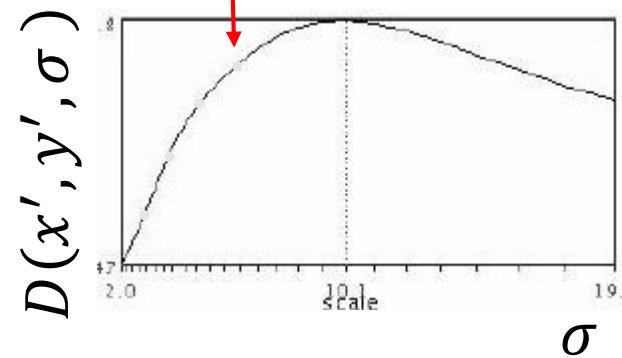
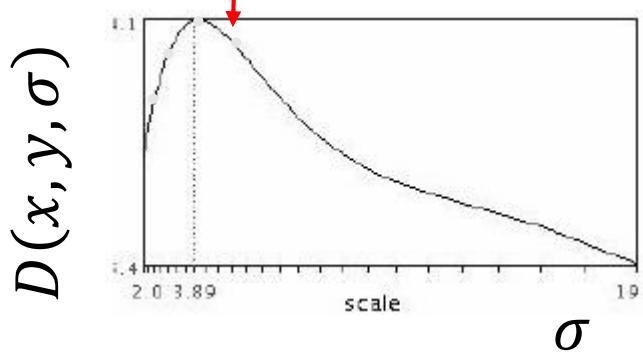
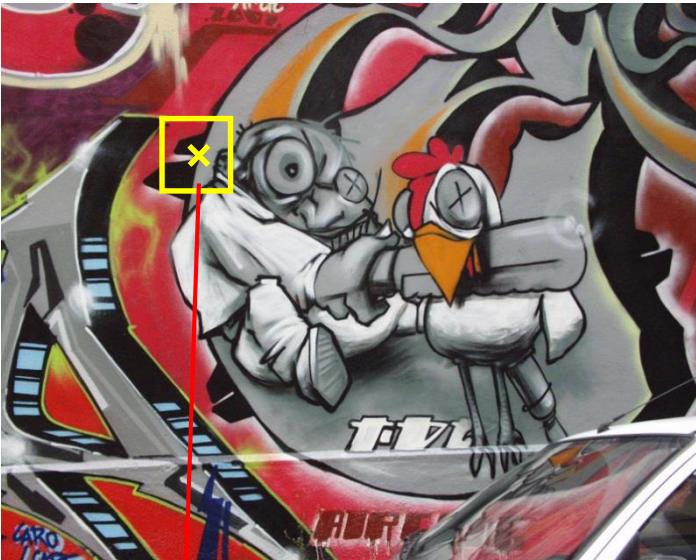
K. Grauman, B. Leibe



K. Grauman, B. Leibe

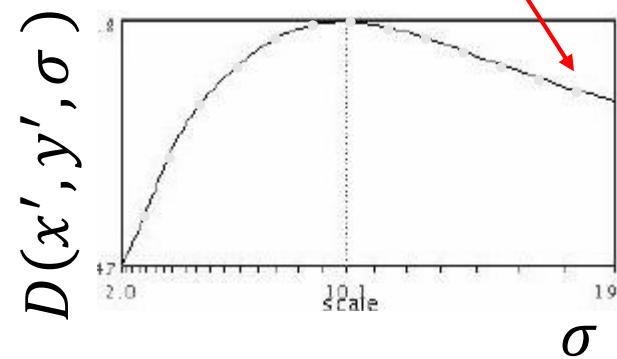
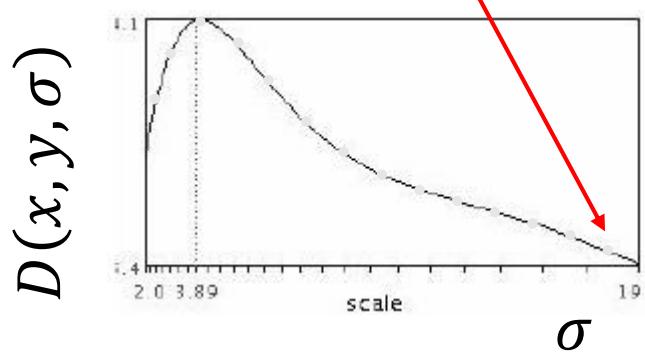
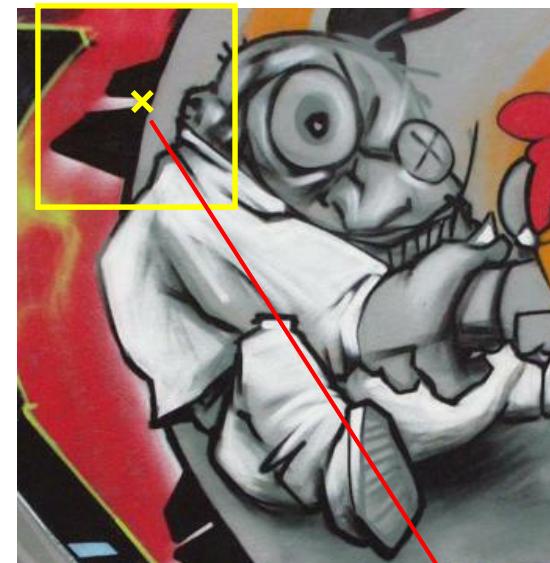
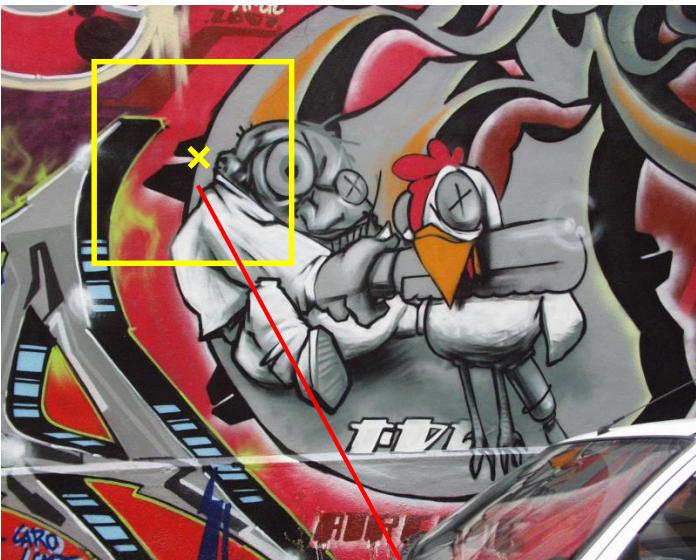
Automatic Scale Selection

Function responses for increasing scale (scale signature)



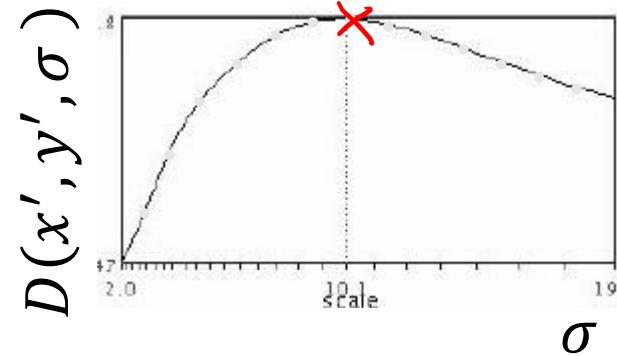
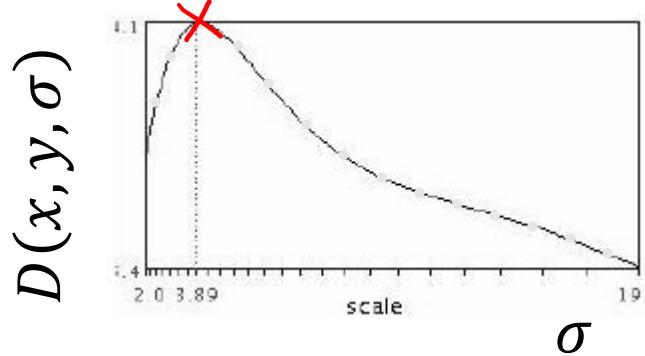
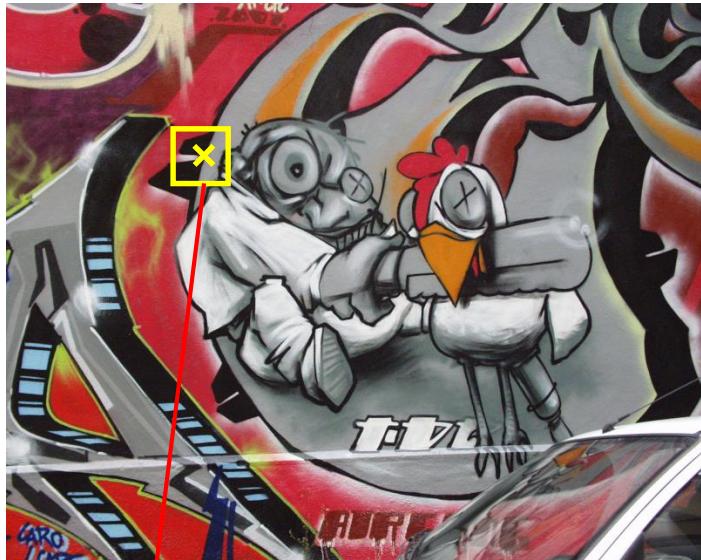
Automatic Scale Selection

Function responses for increasing scale (scale signature)



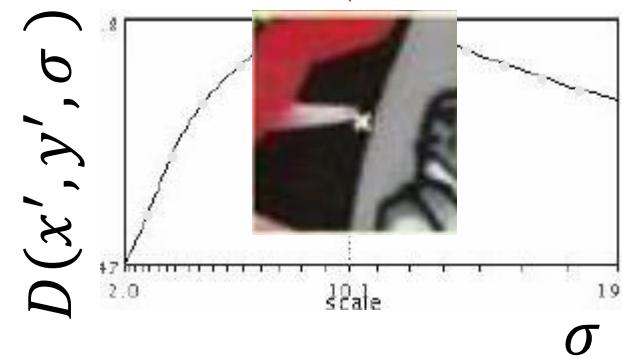
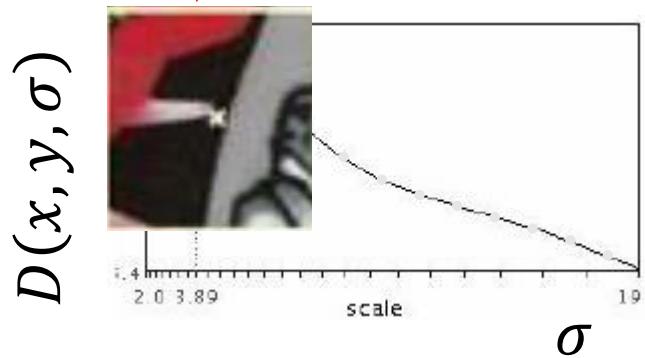
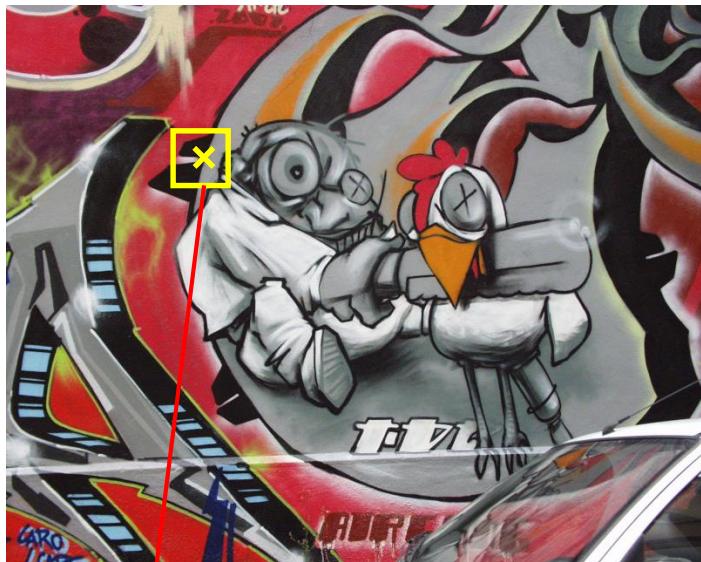
Automatic Scale Selection

Function responses for increasing scale (scale signature)



Automatic Scale Selection

Function responses for increasing scale (scale signature)



The issue



$$D(x, y, \sigma) = D(x', y', \sigma')$$

The intrinsic scale of a keypoint can be identified as a local maxima in the scale space

The approach is indeed general and different functions responses could be considered in principle (e.g. naively the average intensity)

Scale Invariance

- To each keypoint (r, c) we associate the scale $\hat{\sigma}$ of the scale-space corresponding to the local maxima
- The descriptor is computed from the image in the selected scale $L(\cdot, \cdot, \hat{\sigma})$
- **This provides scale-invariance** to the SIFT descriptor

SIFT Keypoint Detector: Lowe ('99)

In particular the following operations are performed:

- Fitting a 3D quadratic function in r, c, σ to interpolate the location of the maximum in the scale-space. This associates to each extrema the 3D-fitted location $(\hat{r}, \hat{c}, \hat{\sigma})$
- Remove low-contrast features by thresholding $D(\hat{r}, \hat{c}, \hat{\sigma})$, e.g.,

$$|D(\hat{r}, \hat{c}, \hat{\sigma})| < 0.3$$

- Remove edges responses, preserving only pixels where D has two large eigenvalues of the Hessian Matrix

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

It is possible to follow an approach similar to Harris detector to avoid computing the SVD.

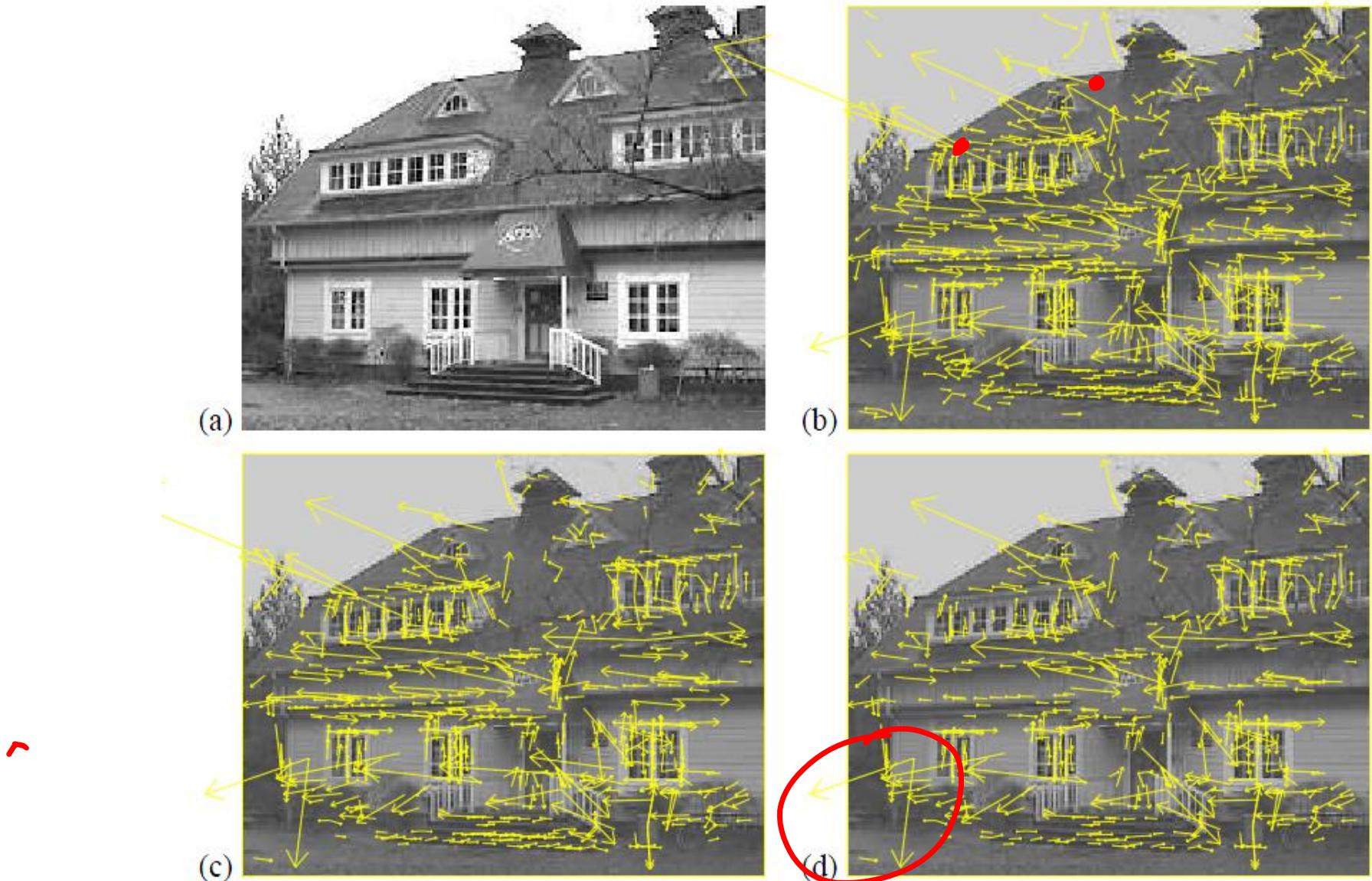


Figure 5: This figure shows the stages of keypoint selection. (a) The 233x189 pixel original image. (b) The initial 832 keypoints locations at maxima and minima of the difference-of-Gaussian function. Keypoints are displayed as vectors indicating scale, orientation, and location. (c) After applying a threshold on minimum contrast, 729 keypoints remain. (d) The final 536 keypoints that remain following an additional threshold on ratio of principal curvatures.

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

Vectors indicate scale, orientation and location.

Scale Invariance

The features are built from the same pyramid used to locate the scale-invariant keypoints

The scale associated to each keypoint (r, c) determines the Gaussian smoothed image, $L(\cdot, \cdot, \sigma)$, that is used to build the descriptor at (r, c)

Thus, each keypoint is associated to a scale of the scale-space

Scale-invariance to the SIFT descriptor is achieved by the **scale-invariance** property of the keypoint

Orientation Assignment

SIFT Scale Invariant Feature Transform [Lowe 2004]

SIFT outline

Scale-space extrema detection: search over all the scales and image locations for potential interest points that are invariant to scale and orientation.

Keypoint localization: At each candidate location, a detailed model is fit to determine location and scale

Orientation assignment: One or more orientations are assigned to each keypoint location based on local image gradient directions.

Keypoint descriptor: The local image gradients are measured at the selected scale in the region around each keypoint

SIFT generates large numbers of features that densely cover the image over the full range of scales and locations

Rotation Invariance: The Basic Idea

Assigning a principal orientation for each keypoint

Each descriptor can be represented relative to this orientation

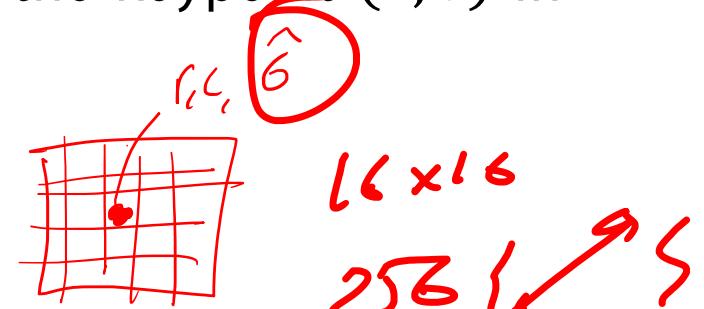
This yields invariance with respect to image rotations

How to Assign an Orientation to Each Keypoint?

Goal: compute the principal orientation in a neighborhood of the keypoint (r, c) in $L(\cdot, \cdot, \hat{\sigma})$ (at the selected scale)

1. For (x, y) in a 16×16 neighborhood of (r, c) compute:

- $\theta(x, y)$ the orientation of the gradient
- $m(x, y)$ the magnitude of the gradient



2. Compute an histogram of the orientations over 36 bins, each bin covering 10 degrees.

3. Weight each orientation by:

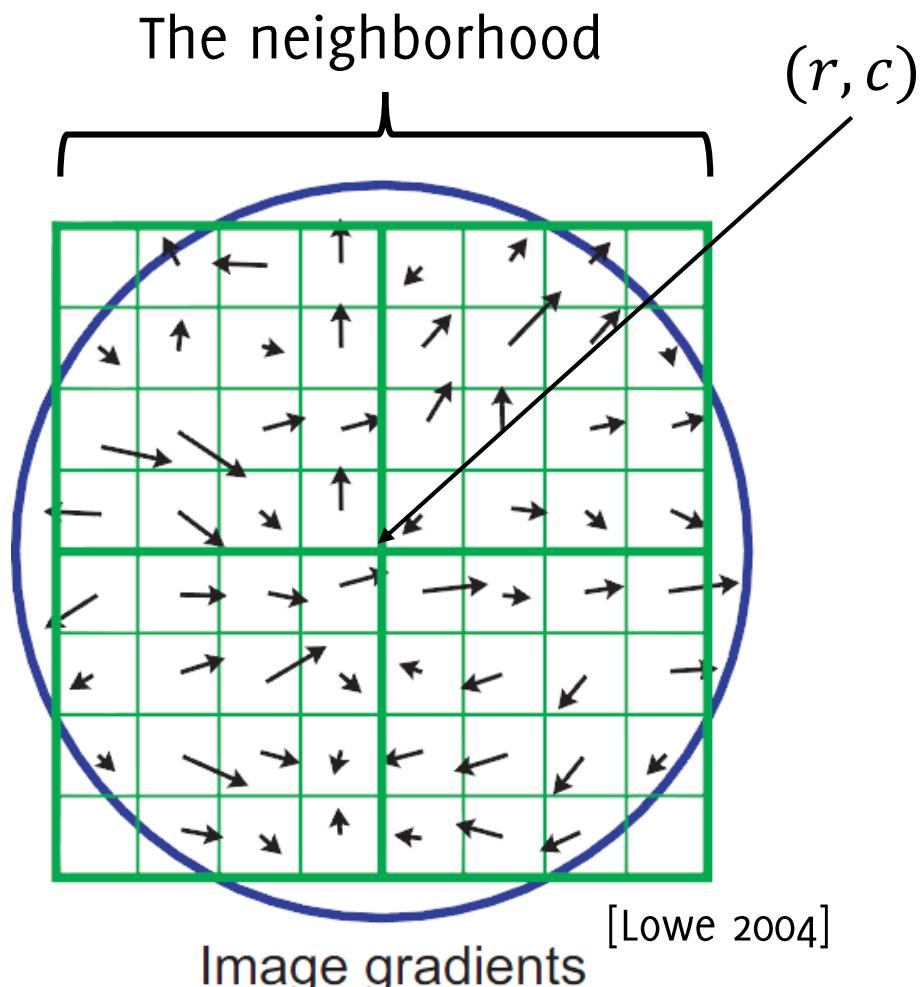
- the gradient magnitude
- a Gaussian weight to give more relevance to estimates that are close to (r, c)

The idea: peaks in the orientation histogram correspond to dominant directions of local gradients



Local Descriptors: Image Gradients

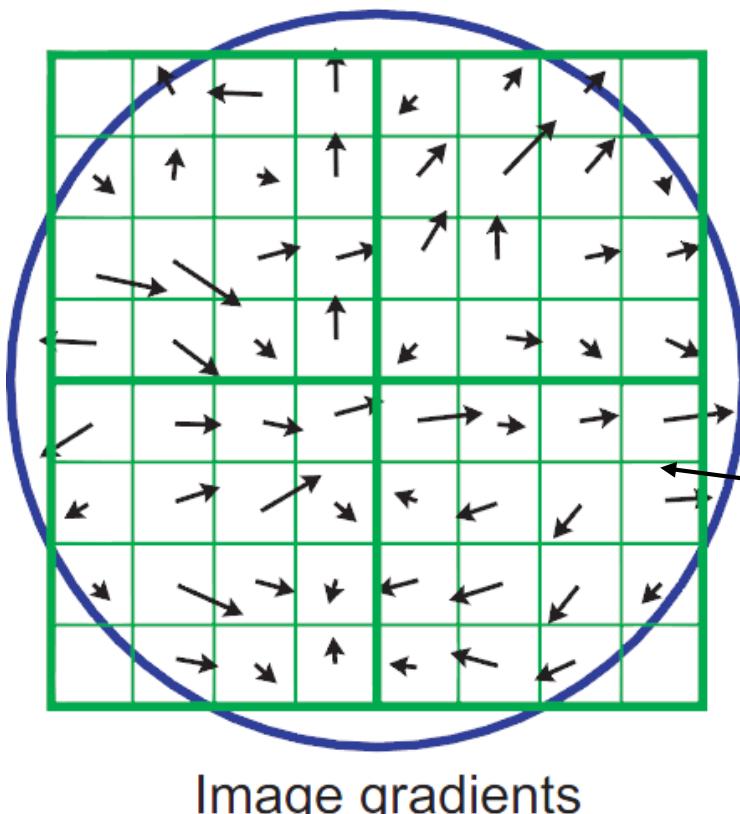
The idea: peaks in the orientation histogram correspond to dominant directions of local gradients



Local Descriptors: the Orientation Histogram

Weight each orientation according to:

- the **gradient magnitude** (orientation at pixels in high-contrast regions are more relevant)
- the **distance from the keypoint** location. This weight is assigned by a Gaussian function having standard deviation $1.5 \hat{\sigma}$, where $\hat{\sigma}$ is the keypoint selected scale

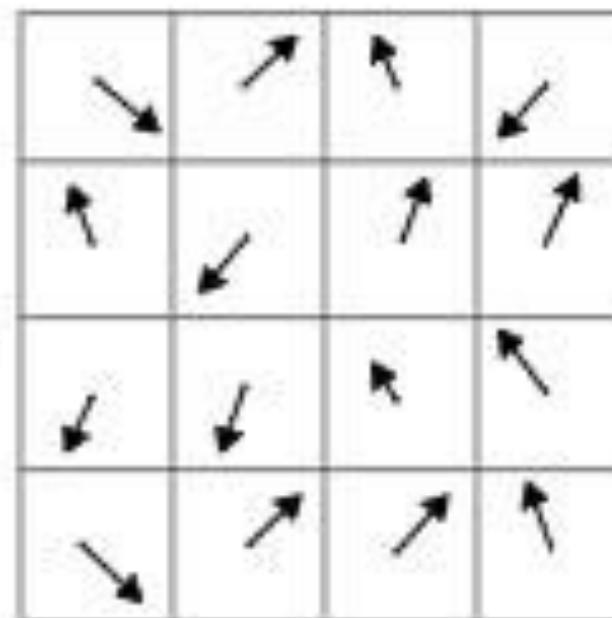


Scaling due to the gradient magnitude is indicated by the length of the arrow. Gaussian weights are indicated by the circle.

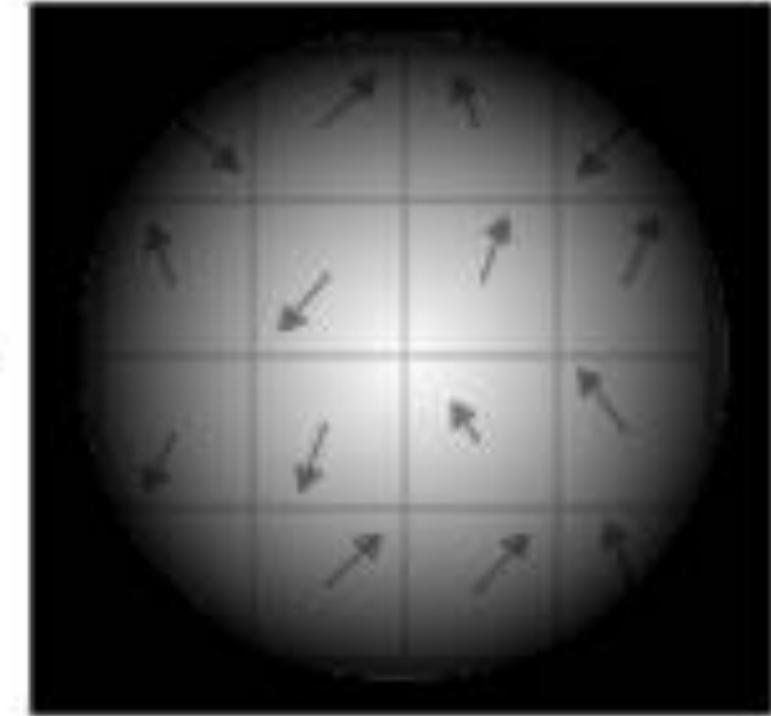
Weighting Scheme



\times

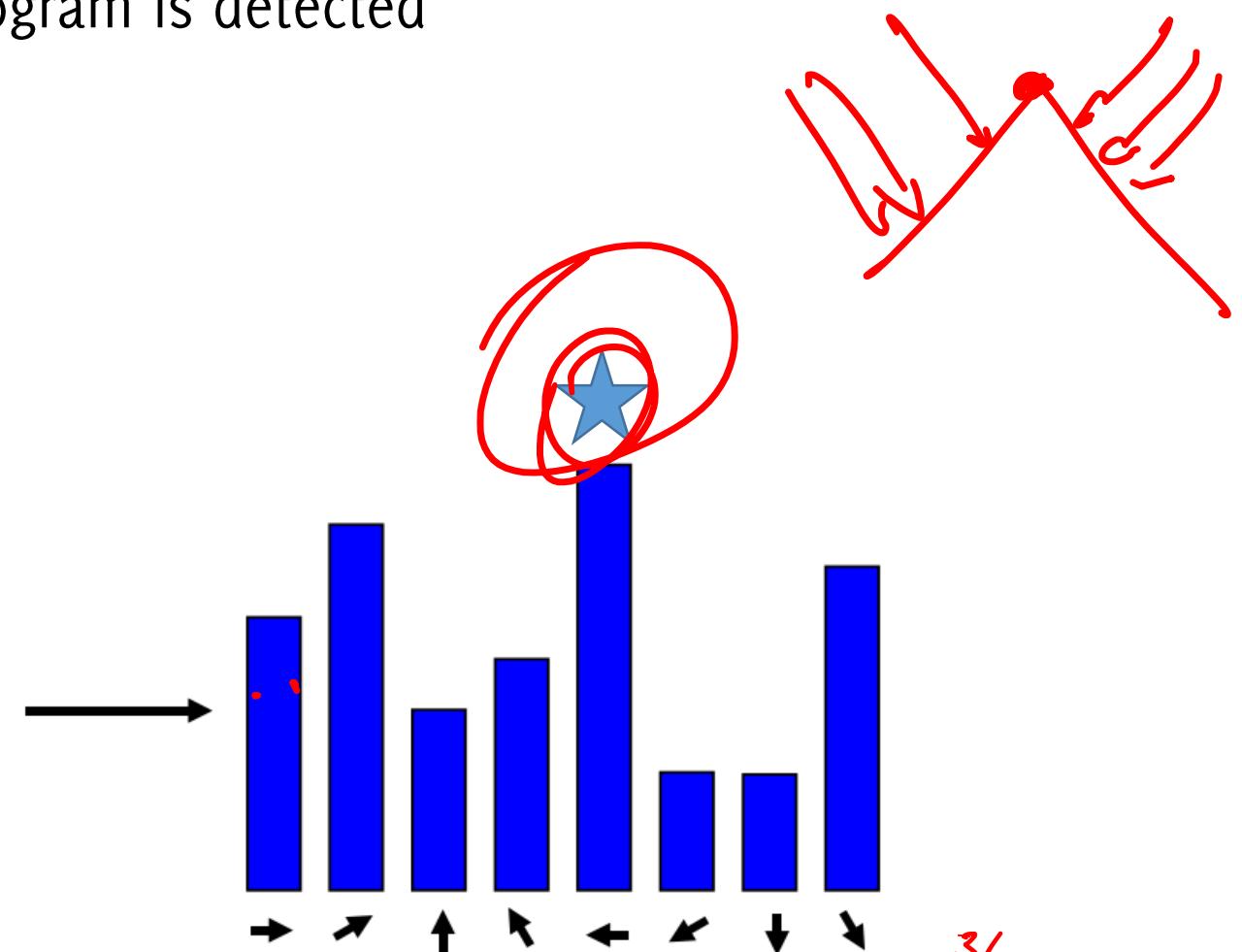
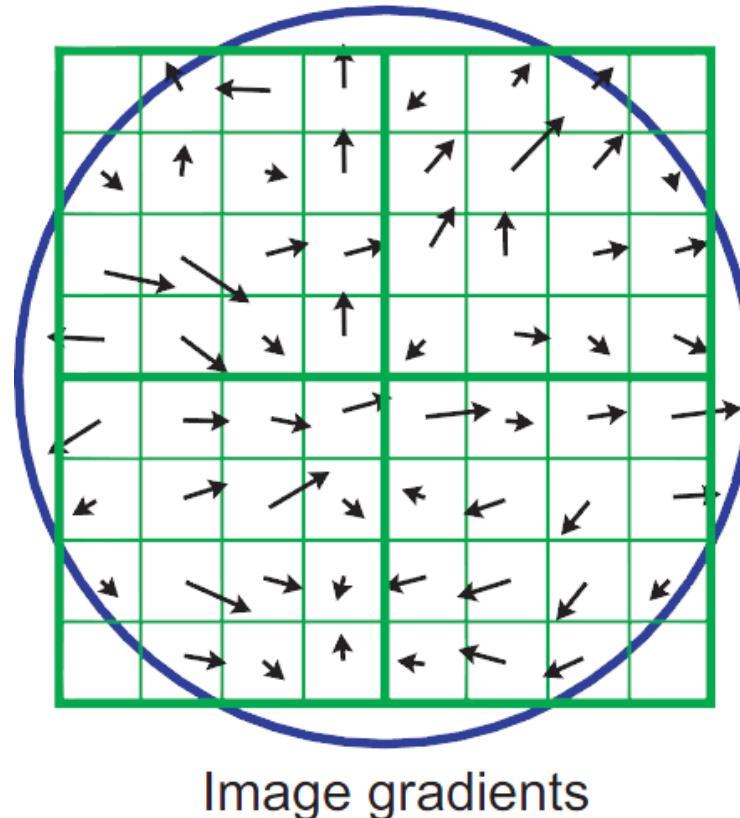


=



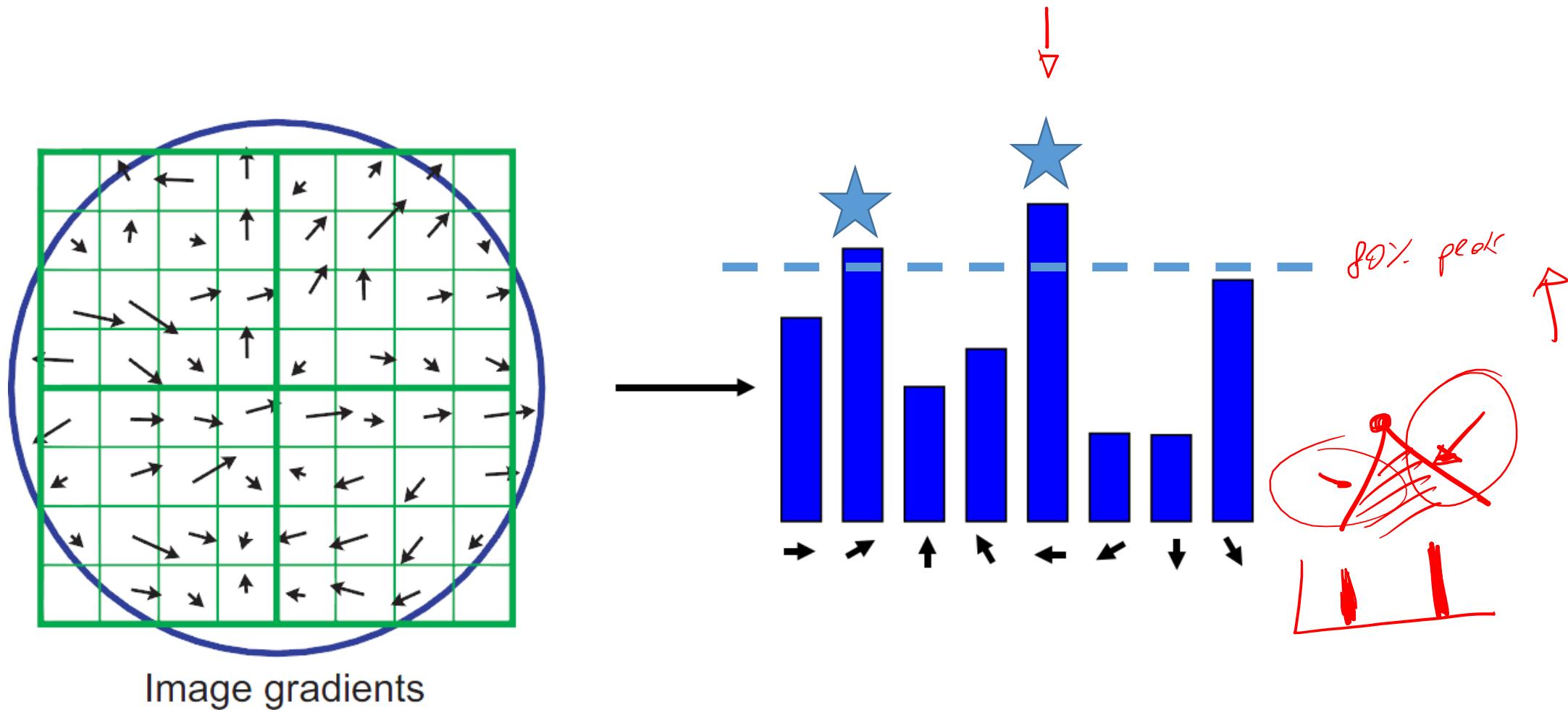
Local Descriptors: Orientation Assignment

The highest peak in the histogram is detected



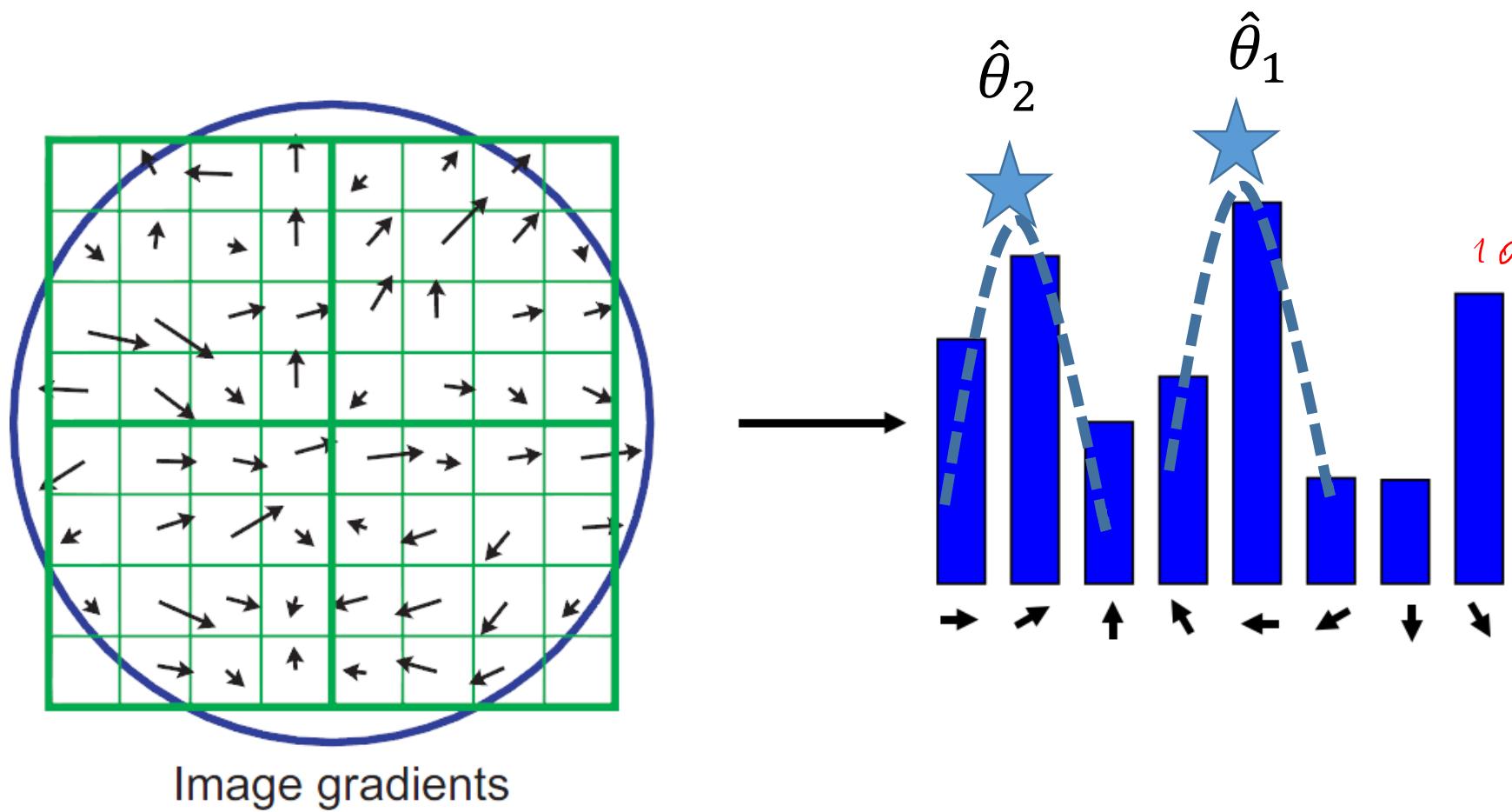
Local Descriptors: Orientation Assignment

The highest peak in the histogram is detected, and then any other local peak that is within 80% of the highest peak is used to also create a keypoint with that orientation



Local Descriptors: Orientation Assignment

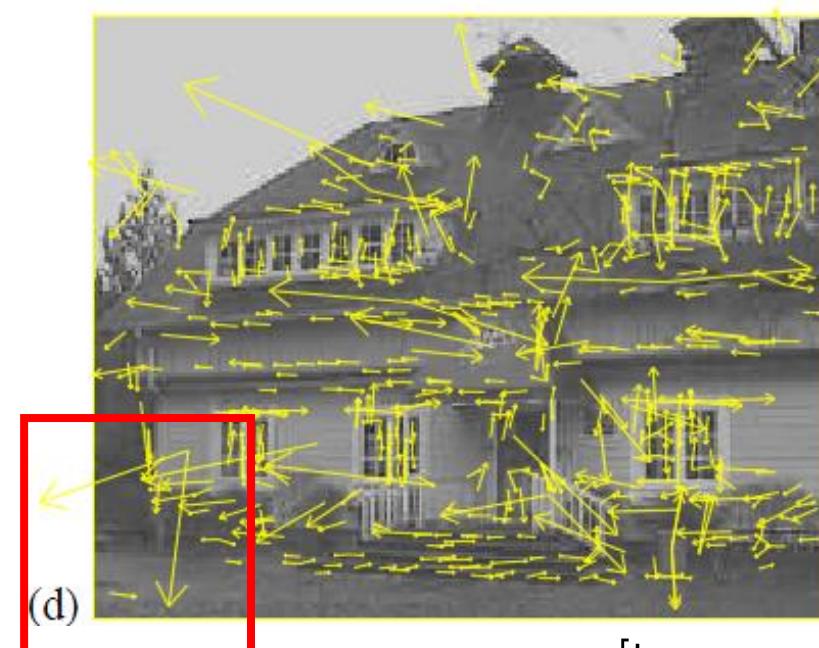
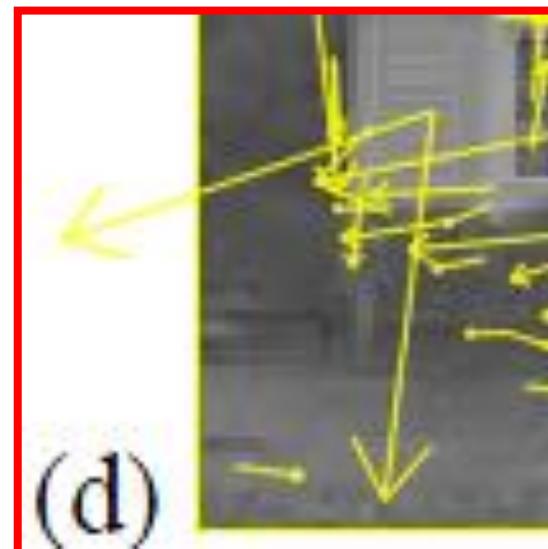
A parabola is fit to the 3 histogram values closest to each peak to interpolate the peak position for better accuracy.



Local Descriptors: Orientation Assignment

Thus, at few locations (about 15% in the experiments in [Lowe 2004]) multiple keypoints might be created at the same location and scale but different orientations

These contribute significantly to the stability of matching.



[Lowe 2004]

› Boracchi



Giacomo Boracchi

Keypoint Descriptor

SIFT Scale Invariant Feature Transform [Lowe 2004]

SIFT Outline

Scale-space extrema detection: search over all the scales and image locations for potential interest points that are invariant to scale and orientation.

Keypoint localization: At each candidate location, a detailed model is fit to determine location and scale

Orientation assignment: One or more orientations are assigned to each keypoint location based on local image gradient directions.

Keypoint descriptor: The local image gradients are measured at the selected scale in the region around each keypoint

SIFT generates large numbers of features that densely cover the image over the full range of scales and locations

The Descriptor [Lowe 2004]

The previous operations have assigned an

- image location \hat{x}, \hat{y}
- scale $\hat{\sigma}$
- orientation $\hat{\theta}$ (and possibly more orientations)

to each keypoint.

Descriptors are built on images transformed w.r.t. the assigned location, orientation, and scale: this assignment provides **invariance with respect to these transformations**.

The SIFT descriptor is then extracted from local image region around each keypoint to be **highly distinctive** and **invariant** as much as possible to other photometric and geometric transformations, such as change in illumination or 3D viewpoint changes.

The SIFT Descriptor

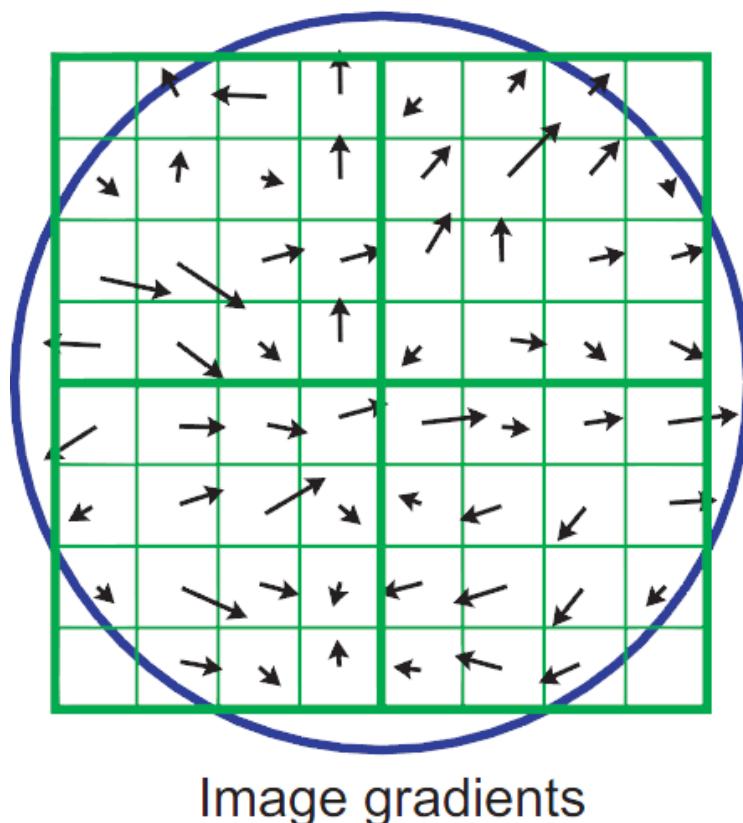
SIFT descriptors are built from the image gradients.

Preprocessing:

- the image gradient magnitudes and orientations are sampled around \hat{x}, \hat{y} , from the layer $\hat{\sigma}$ of the pyramid (i.e. using the selected scale).
- the gradient orientations are rotated relative to $\hat{\theta}$ (i.e., the keypoint orientation).

Local Descriptors: SIFT Descriptor

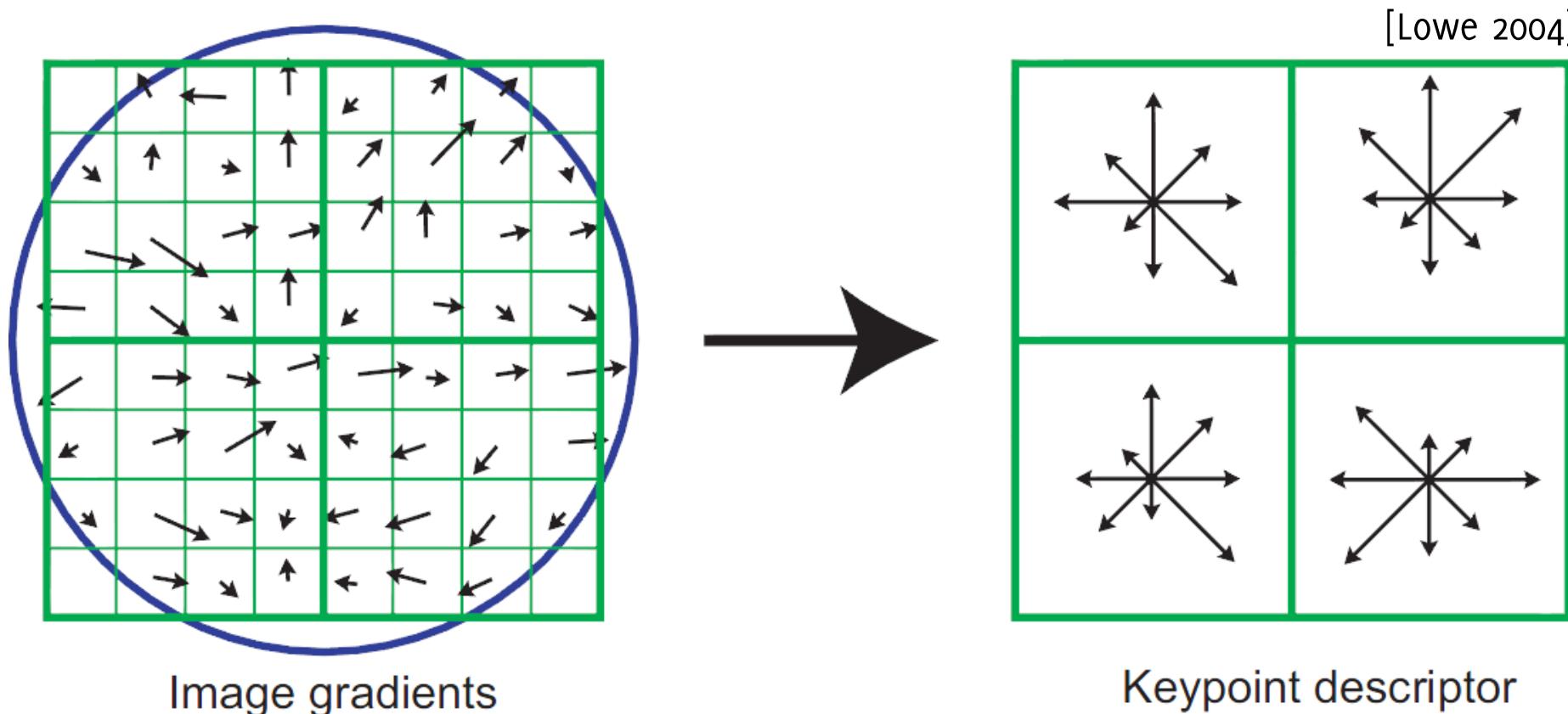
As for orientation assignment, the gradient orientation are weighted w.r.t. the magnitude and the distance from the center (this guarantees robustness to small changes in the position of the window)



The SIFT Descriptor

Four histograms of weighed orientations are created over 8 directions each. The length of each arrow indicates the height of corresponding bin.

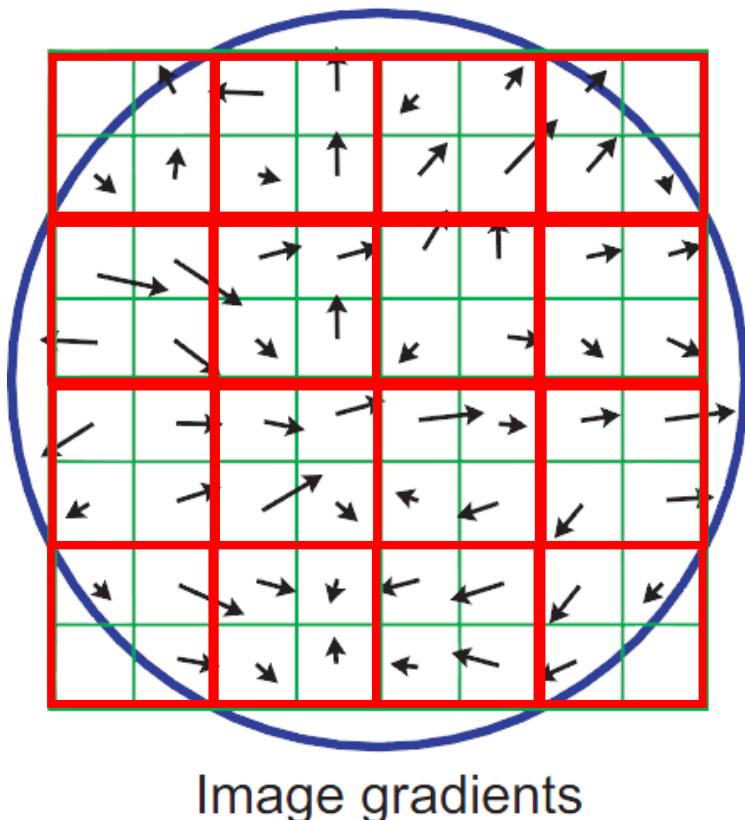
The descriptor is a vector stack of these histograms



The SIFT Descriptor

In the typical implementation, the region is divided in 4×4 regions, each containing an 8-bin histogram.

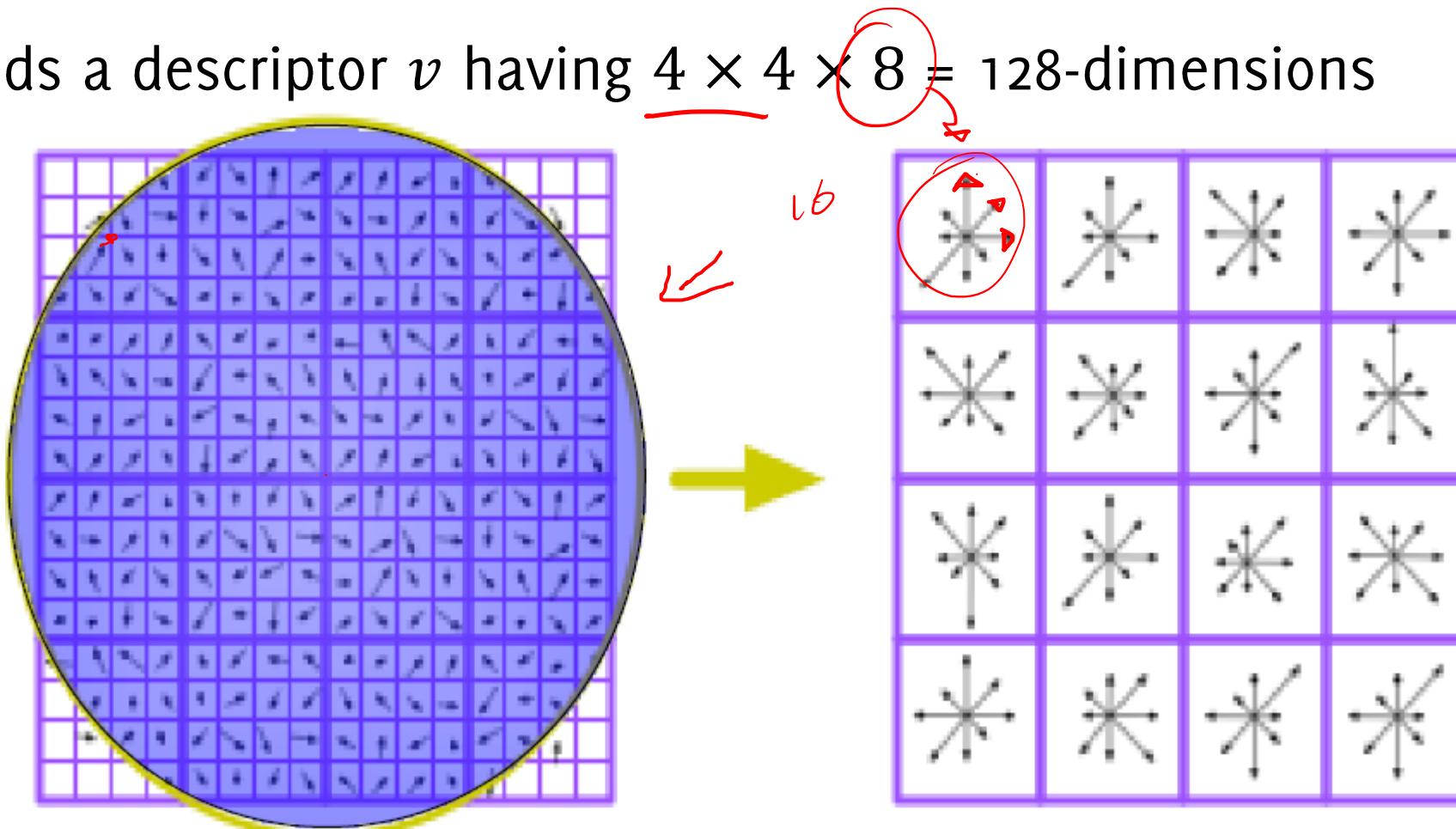
This yields a descriptor ν having $4 \times 4 \times 8 = 128$ -dimensions



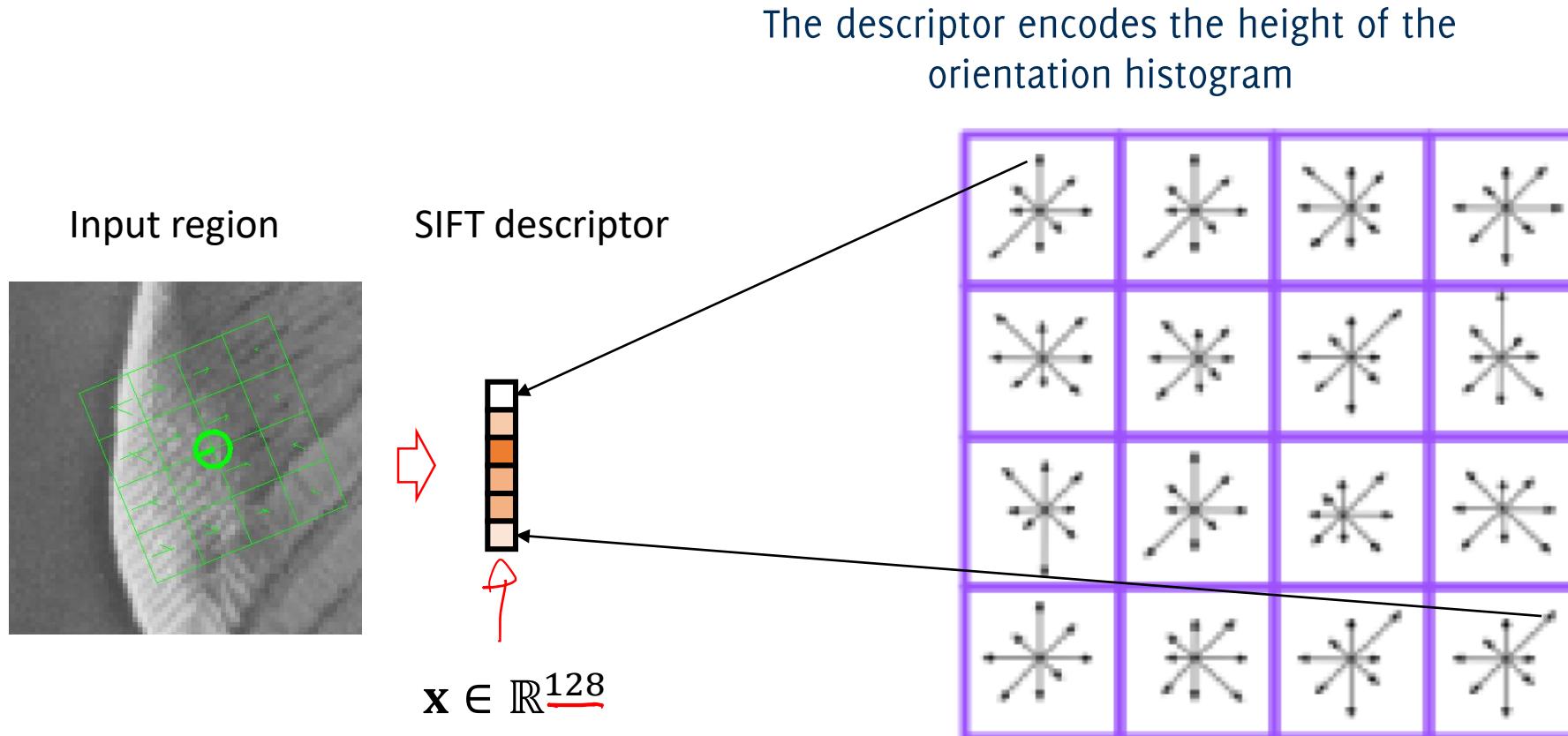
The SIFT Descriptor

In the typical implementation, the region is divided in 4×4 regions, each containing an 8-bin histogram.

This yields a descriptor v having $4 \times 4 \times 8 = 128$ -dimensions

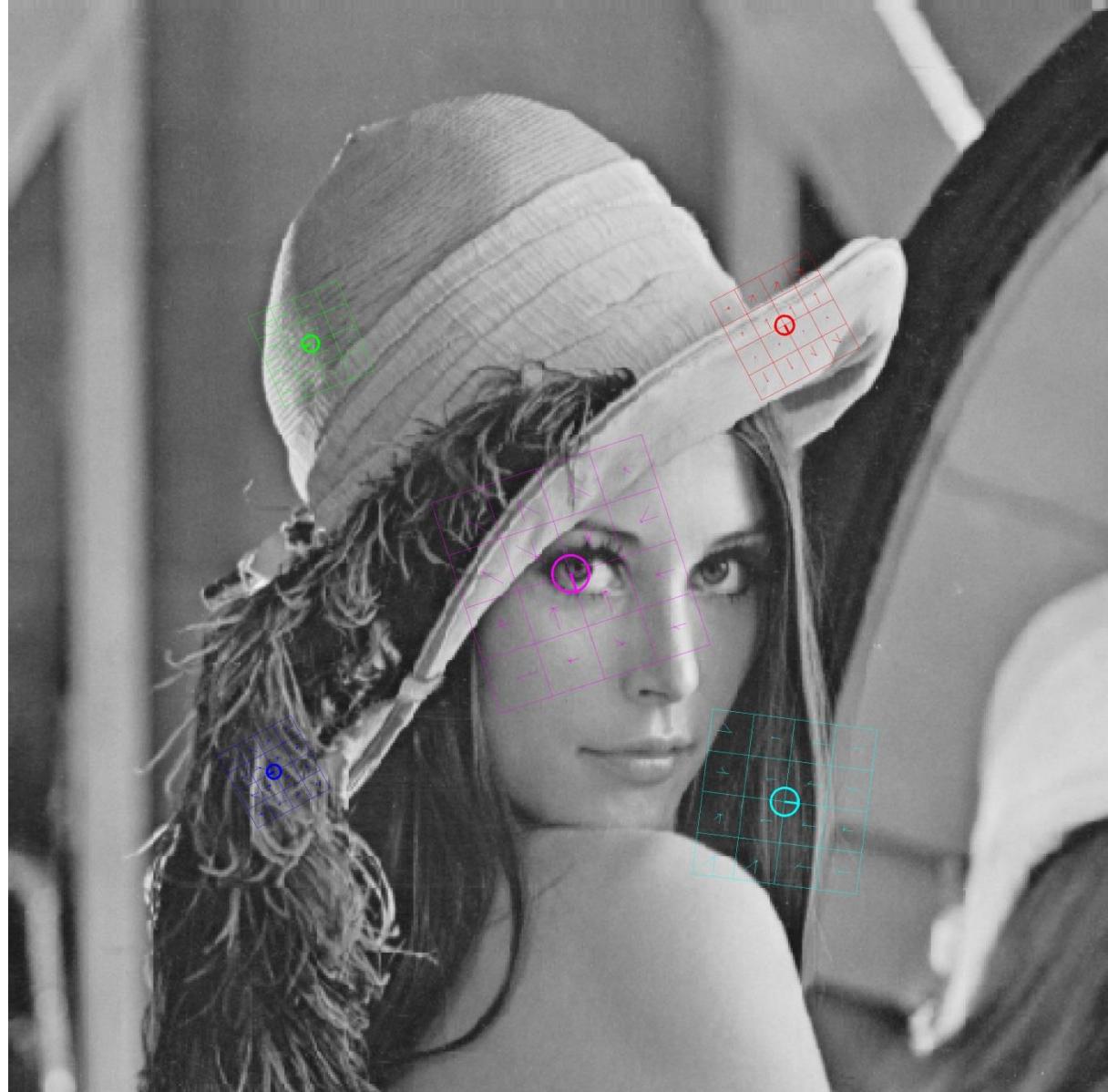


The SIFT Descriptor



An Example

An example of
few SIFT selected
scale and
orientations
(the larger the
square, the
larger the
corresponding
scale in the
scale-space)



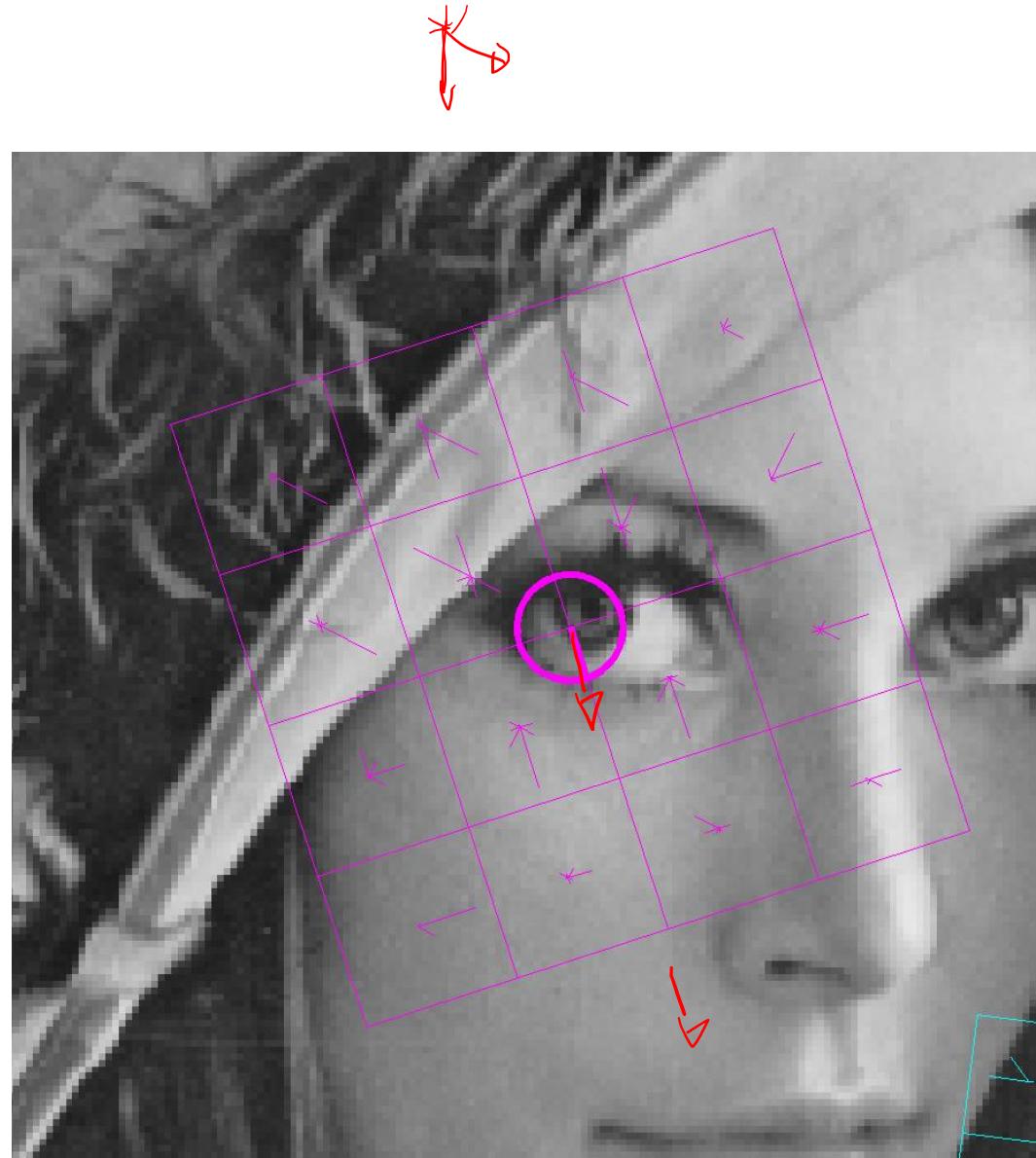
An Example

An example of few SIFT selected scale and orientations

The keypoint was found at an high level of the pyramid, that's why there is a large region around.

Lena' eye is likely to be preserved even by heavy blur in the scale space

Image have been rescaled



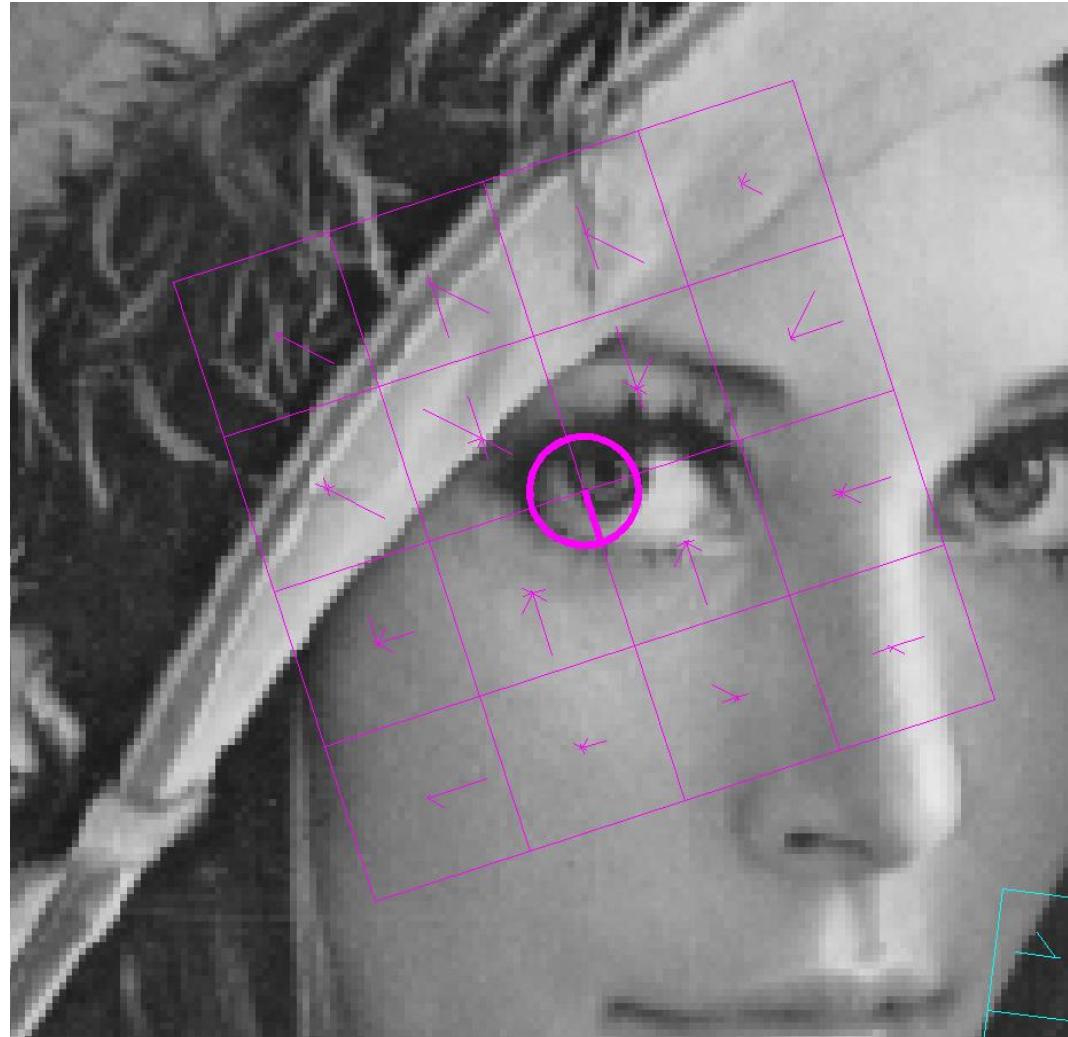
An Example

An example of few SIFT selected scale and orientations

The keypoint was found at an high level of the pyramid, that's why there is a large region around.

Lena' eye is likely to be preserved even by heavy blur in the scale space

Image have been rescaled



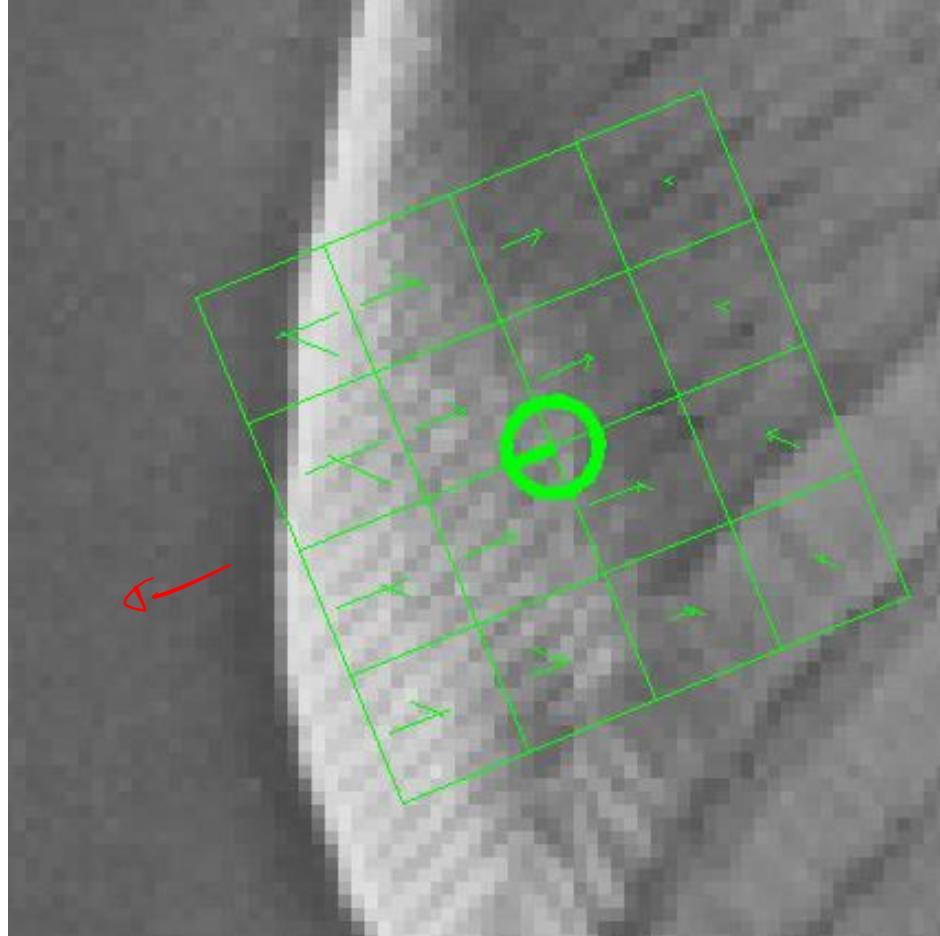
An Example

An example of few SIFT selected scale and orientations

The keypoint was found at a low level of the pyramid, that's why there is a small region around.

Such a texture pattern is likely to be suppressed by blur at lower levels

Image have been rescaled



Robustness to Illumination Changes

SIFT is invariant to affine changes in illumination

- Gradients are themselves invariant to additive shifts, thus SIFT are invariant to «additive illumination changes»
- To achieve invariance to intensity scaling, each descriptor is normalized to yield unitary length i.e. $v \rightarrow \frac{v}{\|v\|_2}$

Nonlinear illumination changes might affect SIFT, introducing gradients having large magnitude.

To increase the robustness to nonlinear illumination changes, the components of v are clipped to 0.2 and then v is normalized again.

Other Descriptors

BRISK, SURF, FREAK, HOG

Other approaches

Lowe has inspired many research works in the following years

Further developments aimed at designing descriptors that are

- more robust to viewpoint changes and artifacts
- easier to extract
- faster to match

SIFT 128 vector *fletig*
 ↓
 36 vector

Example are:

- PCA-SIFT reduces the descriptor vector from 128 to 36 dimension using principal component analysis
- **Speed-up Robust Feature (SURF)**: relies on local gradient histograms computed by the Haar-wavelet that are efficiently computed using integral images (64 dimensional)

SURF

Surf replaces derivative filters used in gradient computation with "flat filters" that assume integer values

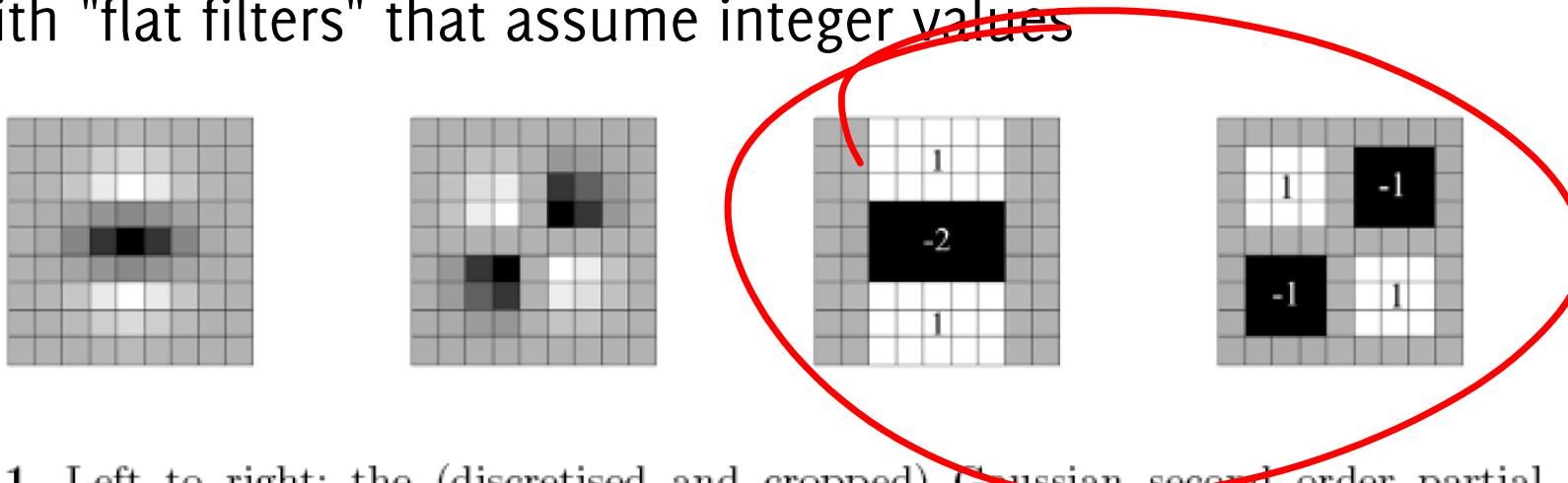


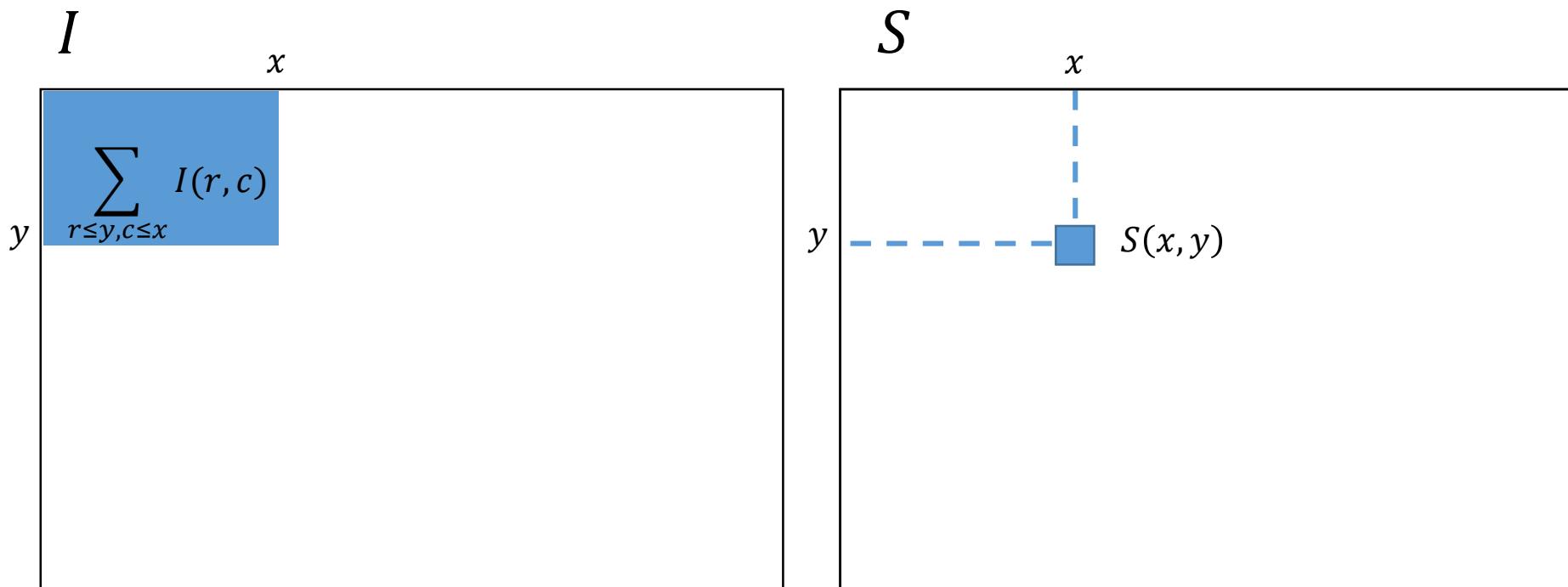
Fig. 1. Left to right: the (discretised and cropped) Gaussian second order partial derivatives in y -direction and xy -direction, and our approximations thereof using box filters. The grey regions are equal to zero.

Convolution against these filters can be efficiently computed by means of the *integral image*

Integral Image

The integral image S is defined from an image I as follows

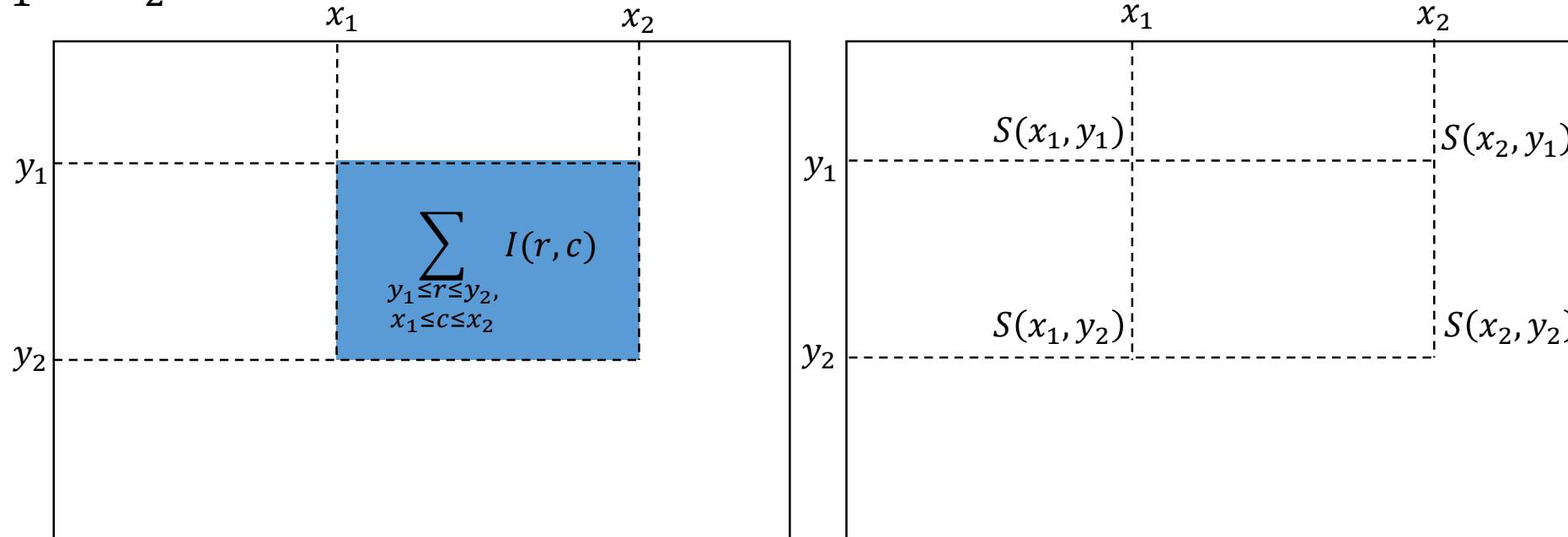
$$S(x, y) = \sum_{r \leq y, c \leq x} I(r, c)$$



Using the Integral Image

The integral image allows fast computation of the sum (average) of any rectangular region in the image

$$\sum_{\substack{y_1 \leq r \leq y_2, \\ x_1 \leq c \leq x_2}} I(r, c) = S(x_2, y_2) - S(x_2, y_1) - S(x_1, y_2) + S(x_1, y_1)$$

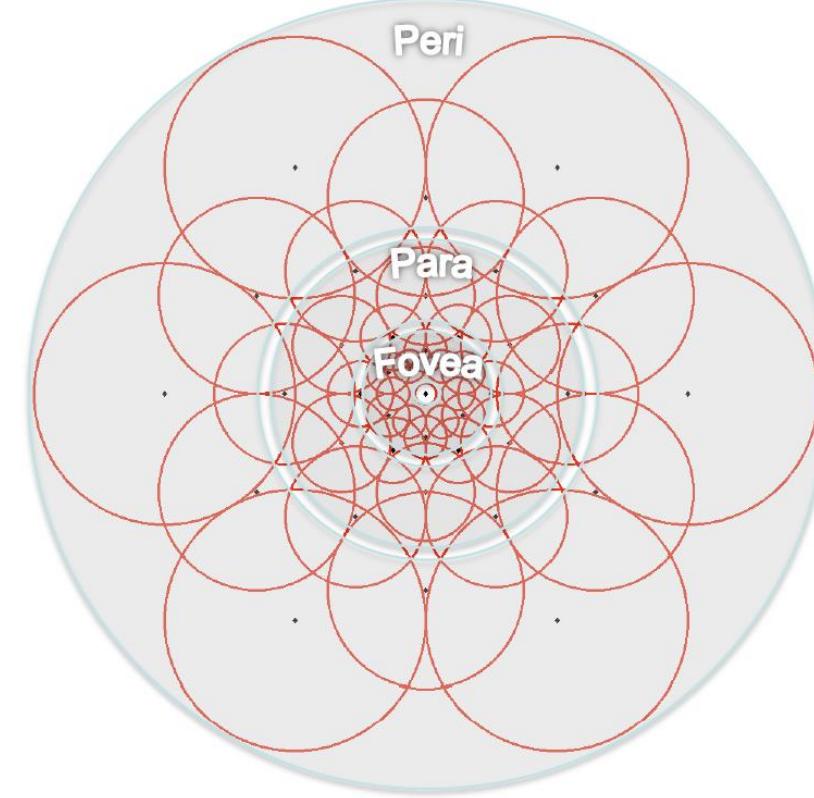


Binary Descriptors

Latest research is devoted to descriptors that are faster to compute, even though less accurate than SIFT.

Freak (fast retina keypoint) is a binary descriptor that encodes the sign of the difference in «receptive fields» around a keypoint

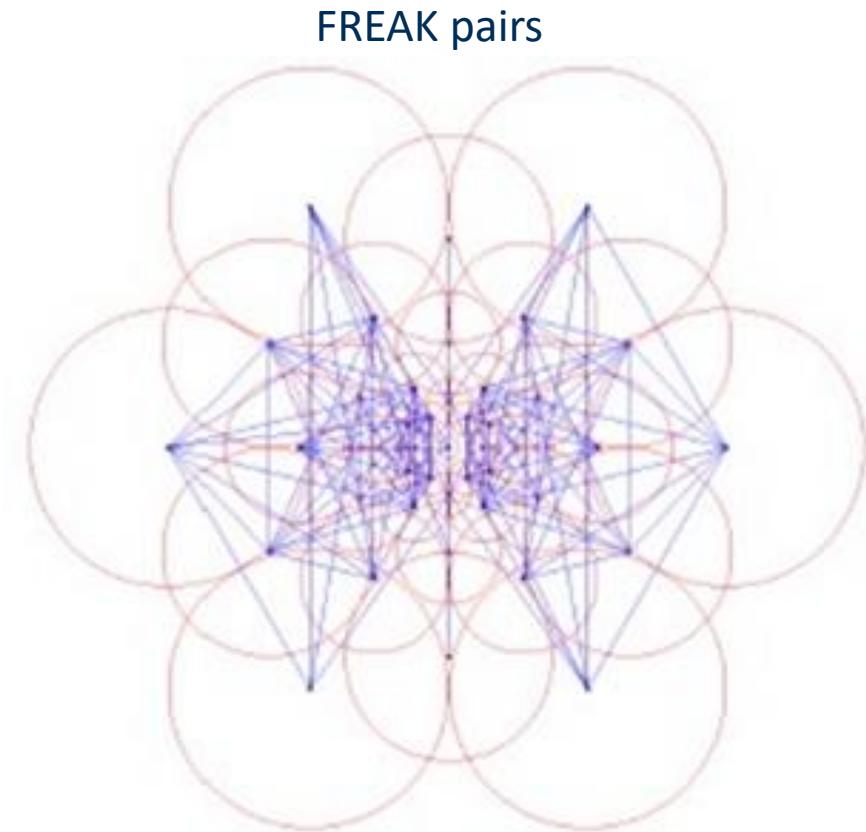
FREAK sampling pattern



Binary Descriptors

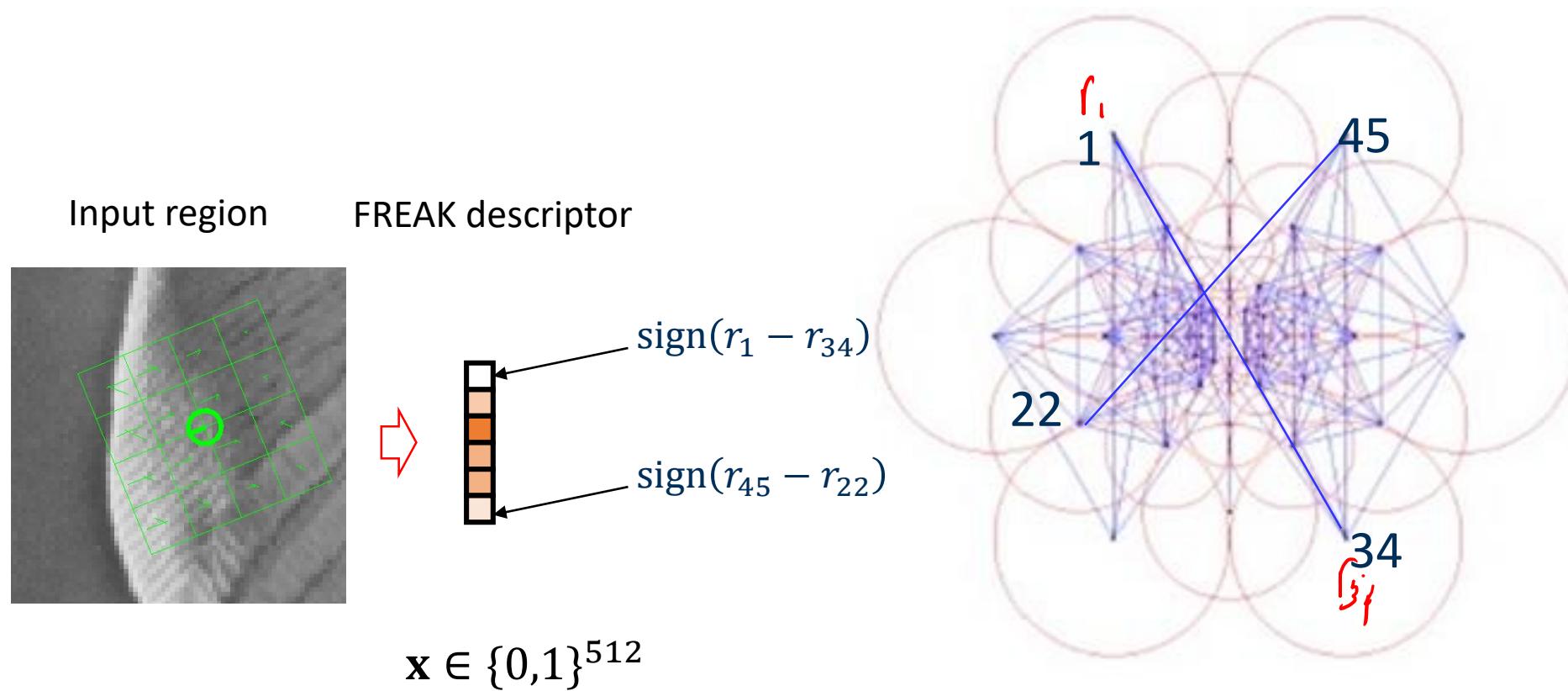
Latest research is devoted to descriptors that are faster to compute, even though less accurate than SIFT.

Freak (fast retina keypoint) is a binary descriptor that encodes the sign of the difference in «receptive fields» around a keypoint



FREAK Descriptor

The descriptor encodes the sign of the difference over pairs of receptive field

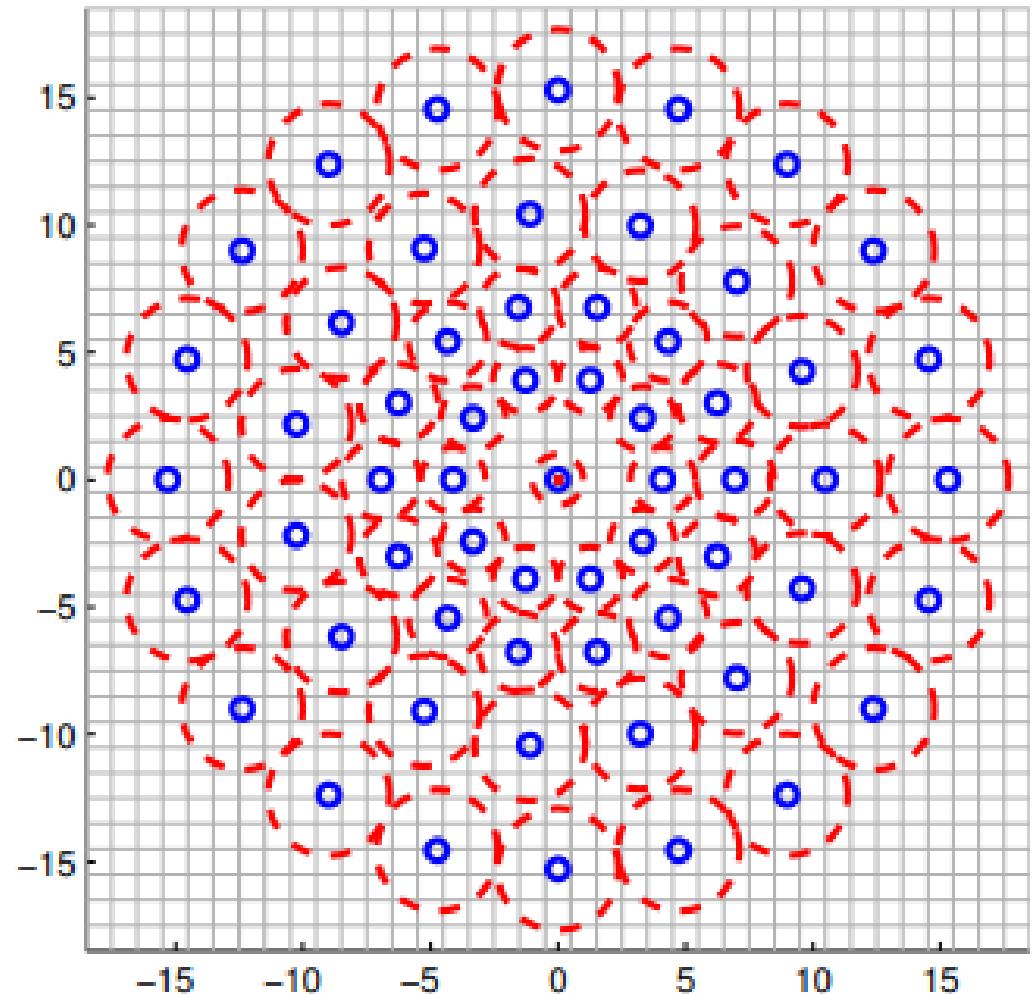


Binary Descriptors

Latest research is devoted to descriptors that are faster to compute, even though less accurate than SIFT.

BRISK (Binary robust invariant scalable keypoints) is a binary descriptor that encodes the sign of the difference in «receptive fields» around a keypoint

BRISK sampling pattern





Histogram Of Oriented Gradients (HOG)

Local object can be characterized rather well by the distribution of local intensity gradients or edge directions, even without precise knowledge of the corresponding gradient or edge positions.

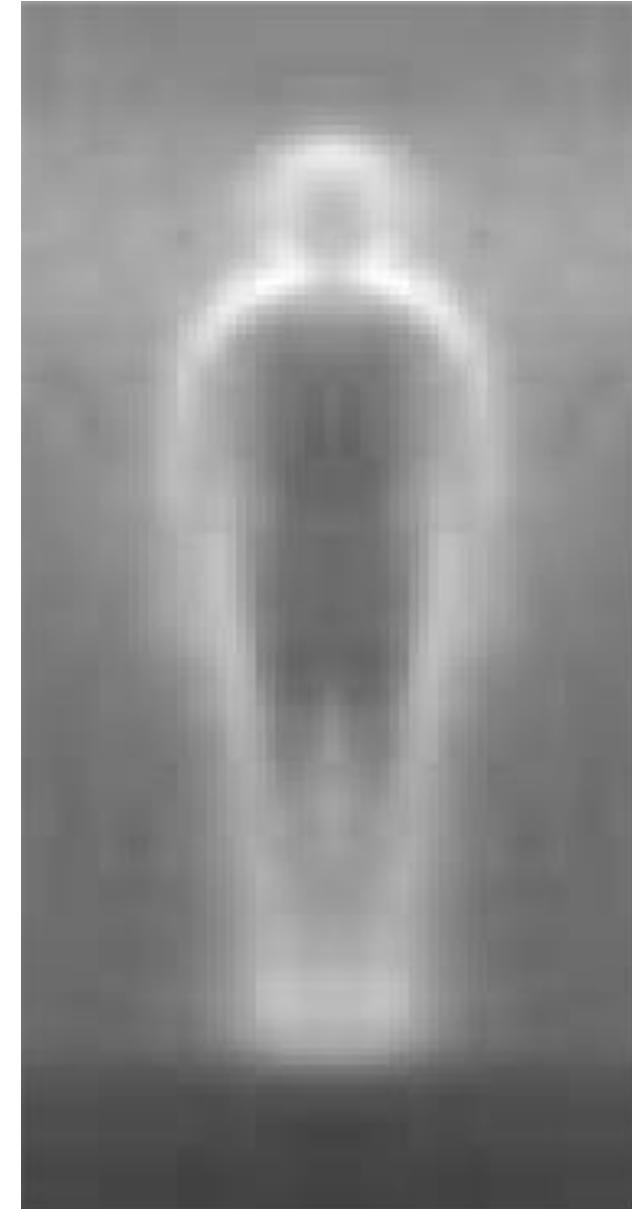
HOG provides

- a **dense descriptor based on gradient directions** (while SIFT is a sparse descriptor)
- overlapping local contrast normalization

Histogram Of Oriented Gradients (HOG)

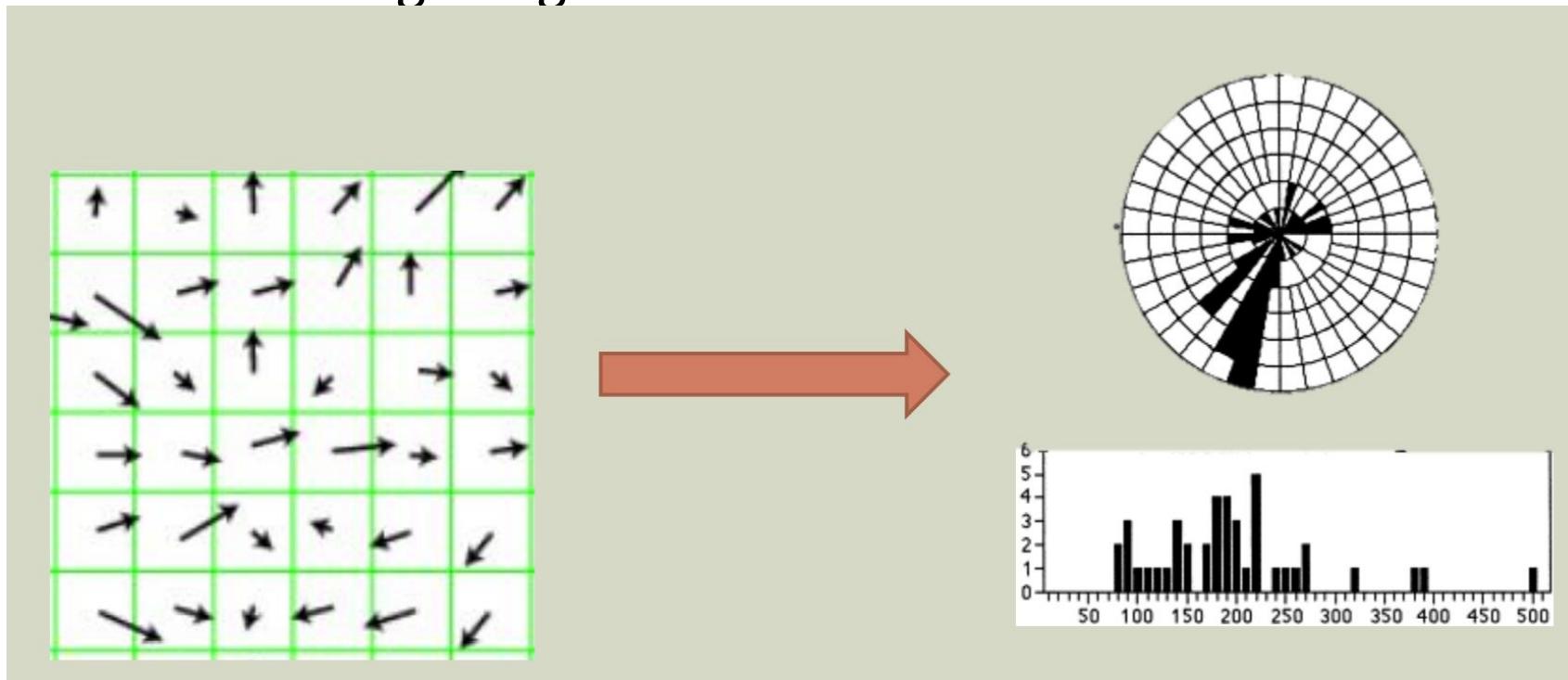
Key idea: Object appearance and shape can often be characterized by the distribution of **local intensity gradients** or edge directions, even without precise knowledge of the corresponding gradient or edge positions

Build a **dense coverage** of descriptors stacking gradient orientations (similar to SIFT) and use these descriptors to **represent the whole image**



HOG Computation

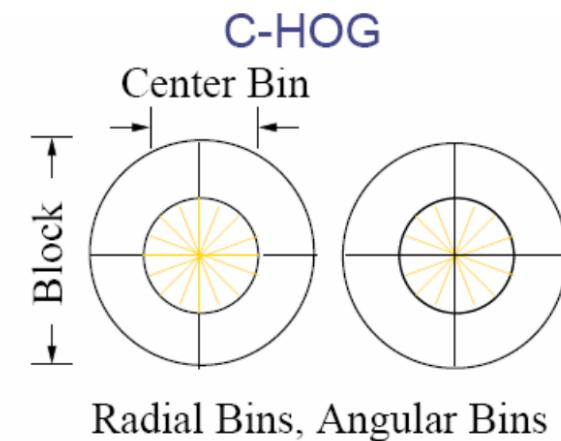
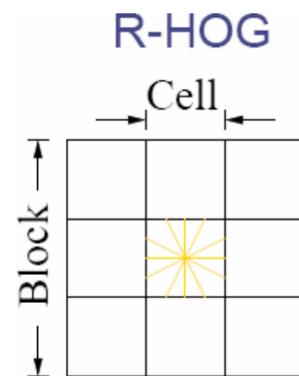
- Divide the image window into small cells (spatial regions over a grid)
Two main block geometries: rectangular (R-HOG) blocks and circular (C-HOG) blocks.
- Compute a weighted local 1-D histogram of gradient directions over the pixels of the cell. Weighting criteria similar to SIFT



HOG Normalization

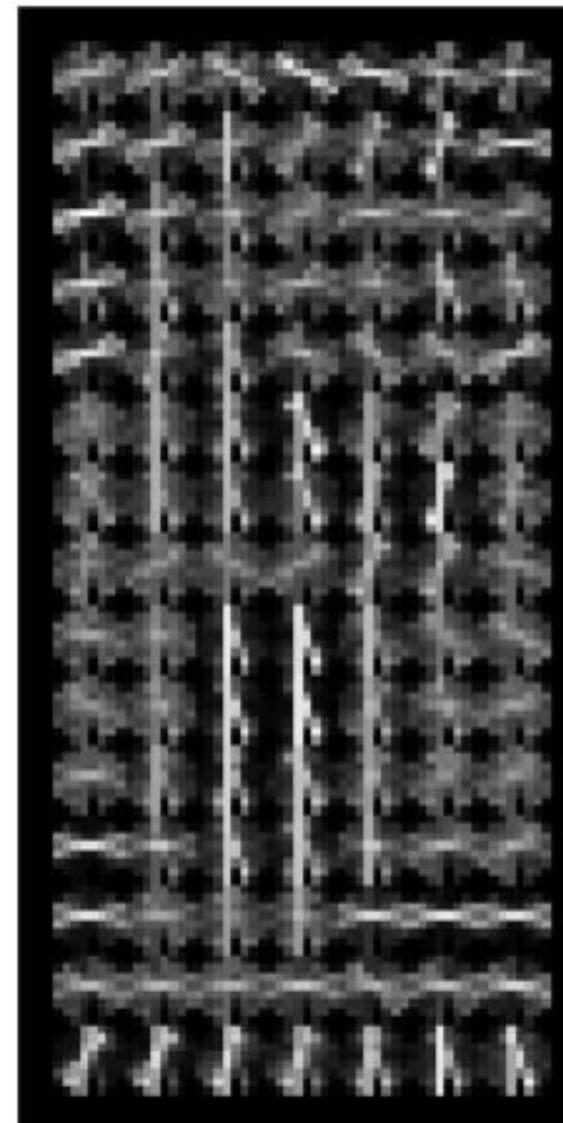
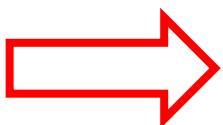
For better invariance to illumination, shadowing, etc., it is also useful to contrast-normalize the local responses before using them.

This can be done by accumulating a measure of local histogram “energy” over somewhat larger spatial regions (“blocks”) and using the results to normalize all of the cells in the block.



Normalization is performed as in SIFT $v \rightarrow \frac{v}{\|v\|_2 + \epsilon}$ being v computer over a block

HOG of an Image



The HOG descriptor
is the concatenation
of all these
histograms

Differences between HOG and SIFT

- HoG is meant to describe entire images. SIFT is used for key point matching
- SIFT histograms are oriented towards the dominant gradient. HoG is not.
- HoG gradients are normalized using neighborhood bins.
- SIFT descriptors use varying scales to compute multiple descriptors, HoG does not.