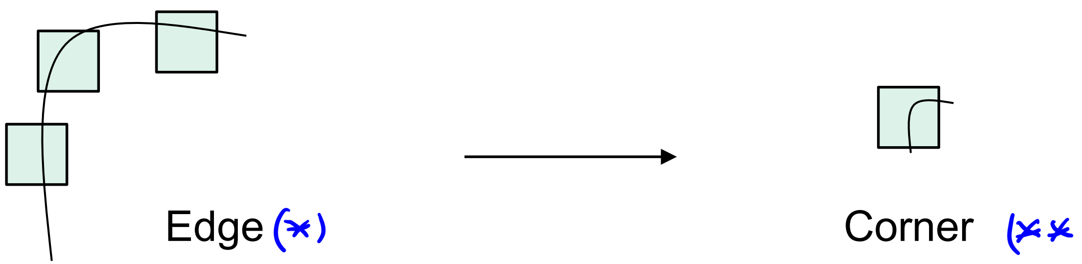
7 Scale Space Representations

7.1 Scale Space Features

Image features can appear similarly on all scales

(**) ⇒ correspondance (*)

Some features appear as such only on a specific scale



In addition to shift / rotation invariance, **scale invariance** is often a desirable feature property

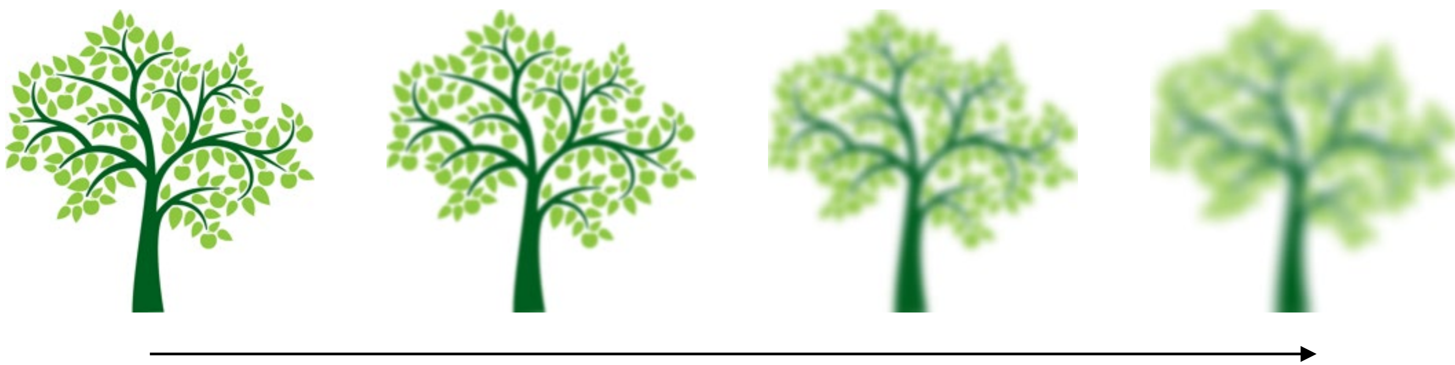
Scale Space Representation

Parametric family of images smoothed by Gaussian filter

Gaussian filter — original input — impulse response of Gaussian LP

$$s[x,y,\sigma] = \underbrace{h_{\mathrm{G}}[x,y,\sigma]}_{\text{Gaussian filter}} * \underbrace{s[x,y]}_{\text{original input}} \qquad \text{with} \qquad h_{\mathrm{G}}[x,y,\sigma] = \frac{1}{2\pi\sigma^2}\mathrm{e}^{-(x^2+y^2)/(2\sigma^2)}$$

$s[x,y,\sigma]$



=> leaves — Scale σ — => tree

Do you want to look at a leaf or at the entire tree?

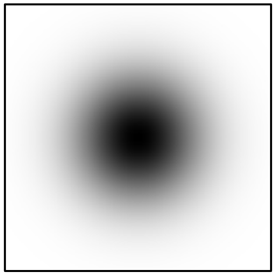
Gaussian Derivatives

Gaussian filter is always positive

- No keypoint detection

Gaussian

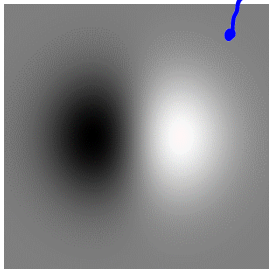
$h_G[x,y]$



horizontal

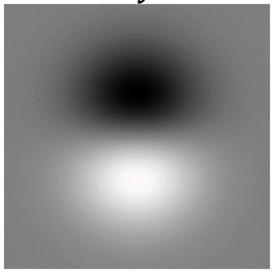
$\frac{\partial h_G[x,y]}{\partial x}$

$\hat{=} 1 \Omega \sigma$

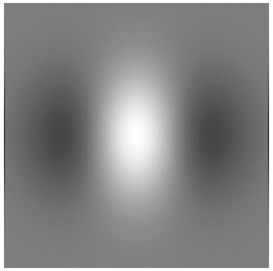


vertical

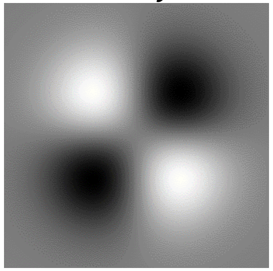
$\frac{\partial h_G[x,y]}{\partial y}$



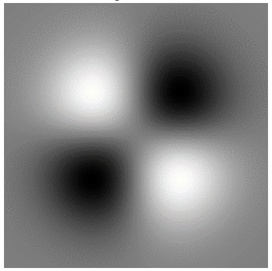
$\frac{\partial^2 h_G[x,y]}{\partial x^2}$



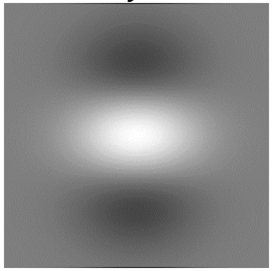
$\frac{\partial^2 h_G[x,y]}{\partial x \partial y}$



$\frac{\partial^2 h_G[x,y]}{\partial y \partial x}$



$\frac{\partial^2 h_G[x,y]}{\partial y^2}$



Hessian Matrix

Square matrix of second order derivatives

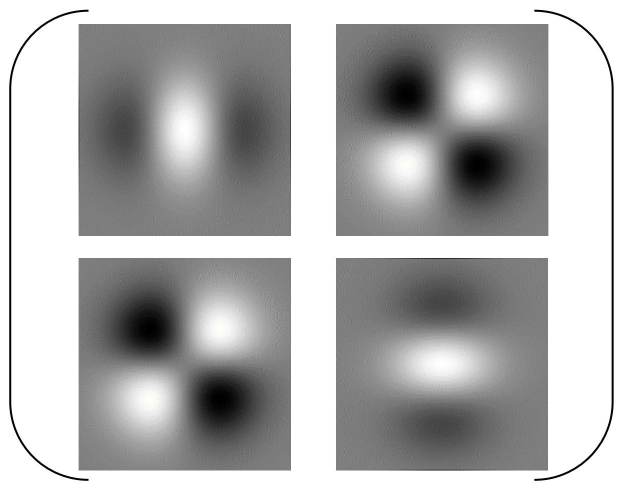
cf. $\underline{M}$ in Harris detector

$$\boldsymbol{H} = \begin{bmatrix} \frac{\partial^2 s[x,y]}{\partial x^2} & \frac{\partial^2 s[x,y]}{\partial x \partial y} \\ \frac{\partial^2 s[x,y]}{\partial x \partial y} & \frac{\partial^2 s[x,y]}{\partial y^2} \end{bmatrix}$$

Special case: **Hessian of Gaussian**

here: s is Gaussian filtered image

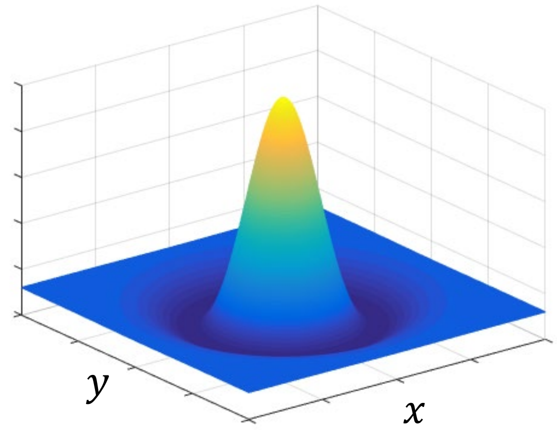
$\boldsymbol{H} =$



7.2 Laplacian of Gaussian (LoG)

Trace of Hessian of Gaussian $\left(\frac{\partial^2 h_G[x,y]}{\partial x^2} + \frac{\partial^2 h_G[x,y]}{\partial y^2}\right) = \nabla^2 s(x,y) = \text{trace}(\underline{H})$

- Sum of main diagonal



Zero-crossings indicate edges

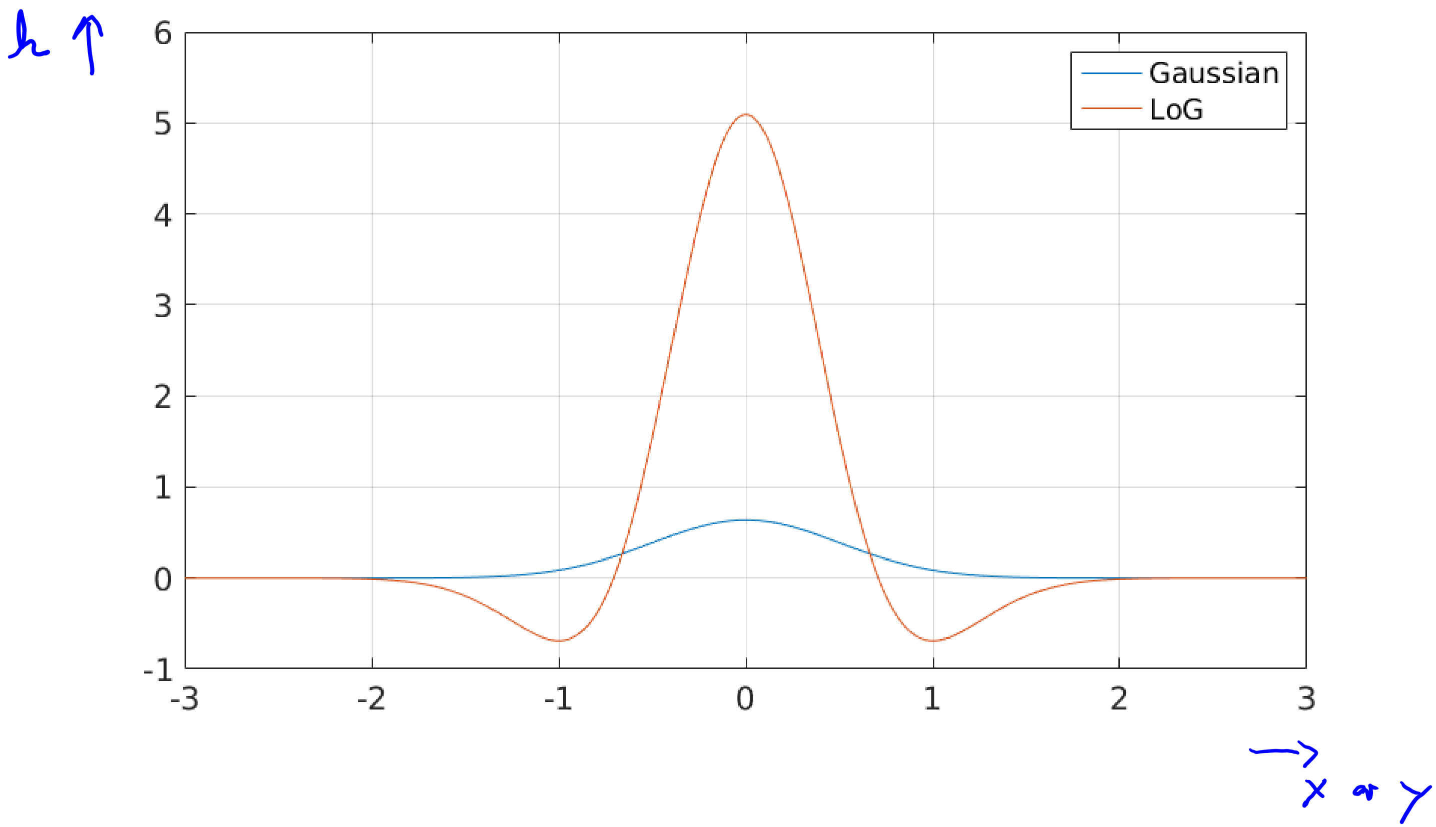
cf. Laplacian edge detector chap. 6-16

Magnitude indicates presence of **blobs**

- Image regions whose properties differ from its surroundings
- Magnitude either positive or negative
- LoG is powerful blob detector

Maximum response if blob “fits” completely under LoG window

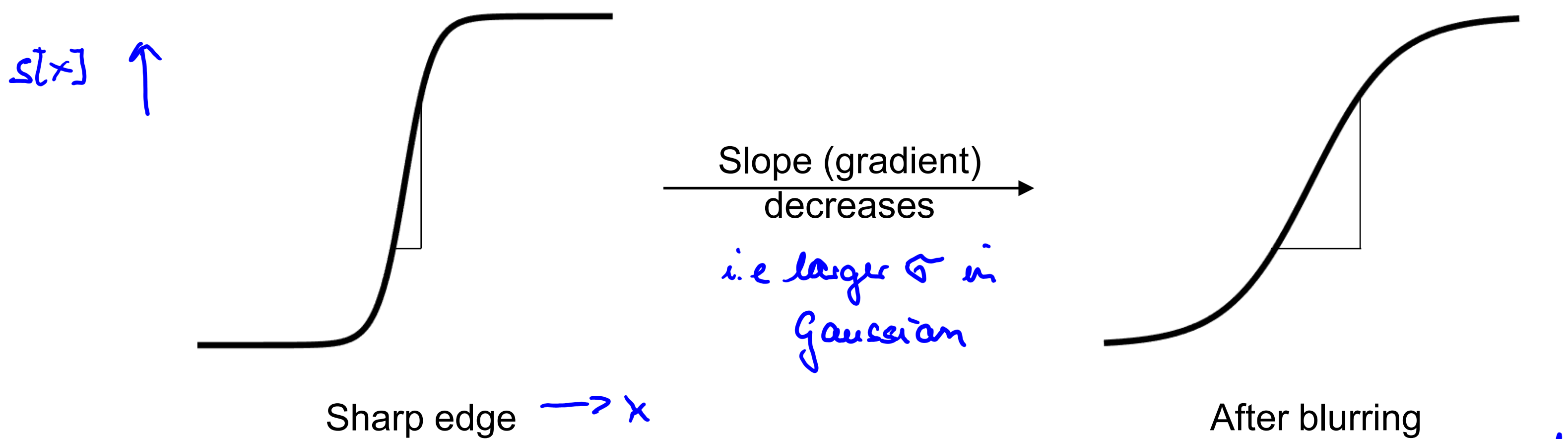
Laplacian of Gaussian (LoG)



Laplacian of Gaussian (LoG)

LoG response is **not** scale invariant

- Gaussian blur smooths contours → lower gradient



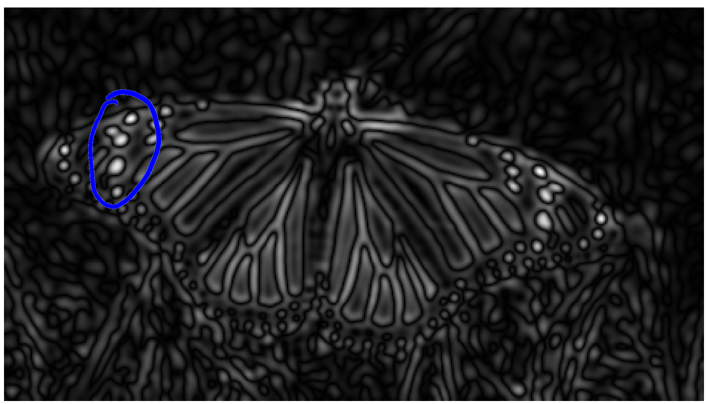
=> smaller gradient

Scale-normalized LoG: $\sigma^2\nabla^2 \overbrace{h_{\mathrm{G}}[x, y, \sigma]}^{\text{Gaussian}}$

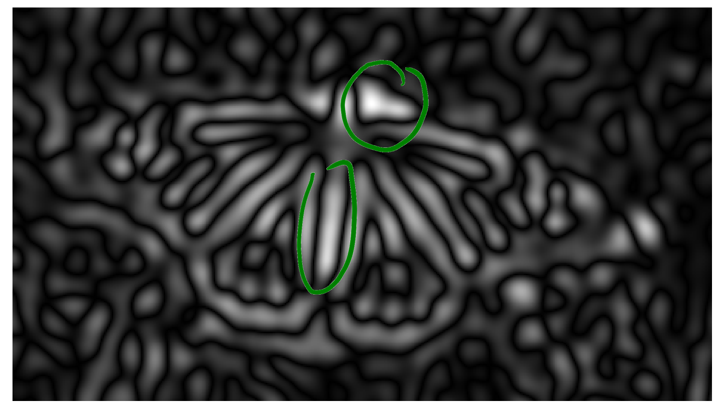
- Multiplication by σ^2 compensates for gradient decrease caused by Gaussian blur

LoG Example

$\sigma = 1$



$\sigma = 3$



$\sigma = 8$

For small (large) scales LoG magnitude highlights small (large) details.

Difference of Gaussians (DoG)

Computing LoG might be inefficient, i.e. computationally too costly

Gaussian kernel is solution to diffusion equation:

$$\frac{\partial h_{\mathrm{G}}[x,y,\sigma]}{\partial\sigma} = \sigma\nabla^2 h_{\mathrm{G}}[x,y,\sigma]$$

Finite difference approximation:

$$\underbrace{\cancel{\sigma}\nabla^2 h_{\mathrm{G}}[x,y,\sigma]} = \frac{\partial h_{\mathrm{G}}[x,y,\sigma]}{\partial\sigma} \approx \underbrace{\frac{h_{\mathrm{G}}[x,y,k\sigma] - h_{\mathrm{G}}[x,y,\sigma]}{k\cancel{\sigma} - \cancel{\sigma}}}$$

Therefore:

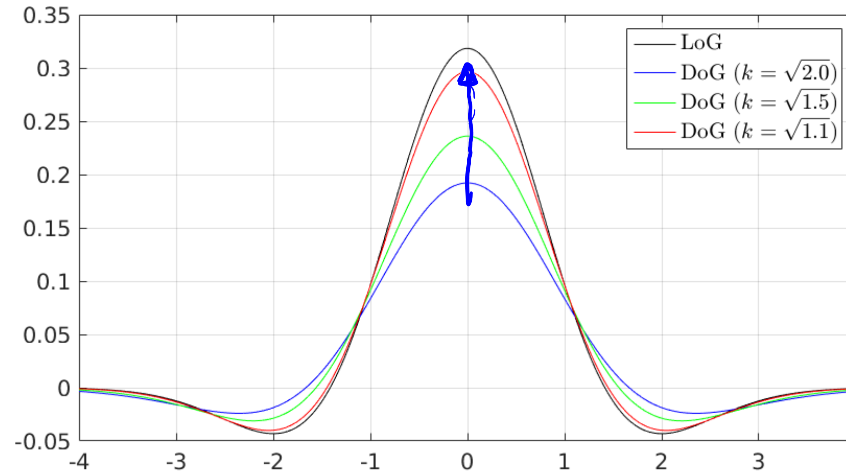
$$\boxed{\mathrm{DoG} = h_{\mathrm{G}}[x,y,k\sigma] - h_{\mathrm{G}}[x,y,\sigma] \approx (k-1)\sigma^2\nabla^2 h_{\mathrm{G}}[x,y,\sigma]}$$

Difference of Gaussians (DoG)

Approximation error → 0

- As k goes to 1

h ↑



→ x

$$\mathrm{DoG} \approx (k-1)\boxed{\sigma^2}\nabla^2 h_{\mathrm{G}}[x,y,\sigma]$$

Advantage: DoG already incorporates $\boxed{\text{scale normalization}}$
Efficient computation

Factor $(k-1)$ is not a problem if k is constant over all scales

- Usually we are interested in peak locations and **relative** magnitudes

7.3 SIFT (David G. Lowe, ICCV 1999)

Scale Invariant Feature Transform

- Patented by University of British Columbia
- Invariant to scale and rotation
- Robust to affine distortions, noise, illumination changes

Highly distinctive features DoG used to detect features; plus new descriptor

- Good for matching, recognition, etc.

SIFT

Consists of feature **detector** and feature **descriptor**

- They are independent (e.g. SIFT descriptor can be used for Harris corners) (or any other)

Four stages:

1) Scale Space Extrema Detection => blob detection, find important points
2) Keypoint Localization (across scale)
3) Orientation Assignment (use dominant orientation)
4) Keypoint Descriptor (create binary description)

1) Scale Space Extrema Detection

use idea of DoG (instead of LoG)

$$\boxed{k = 2^{1/s}}$$

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

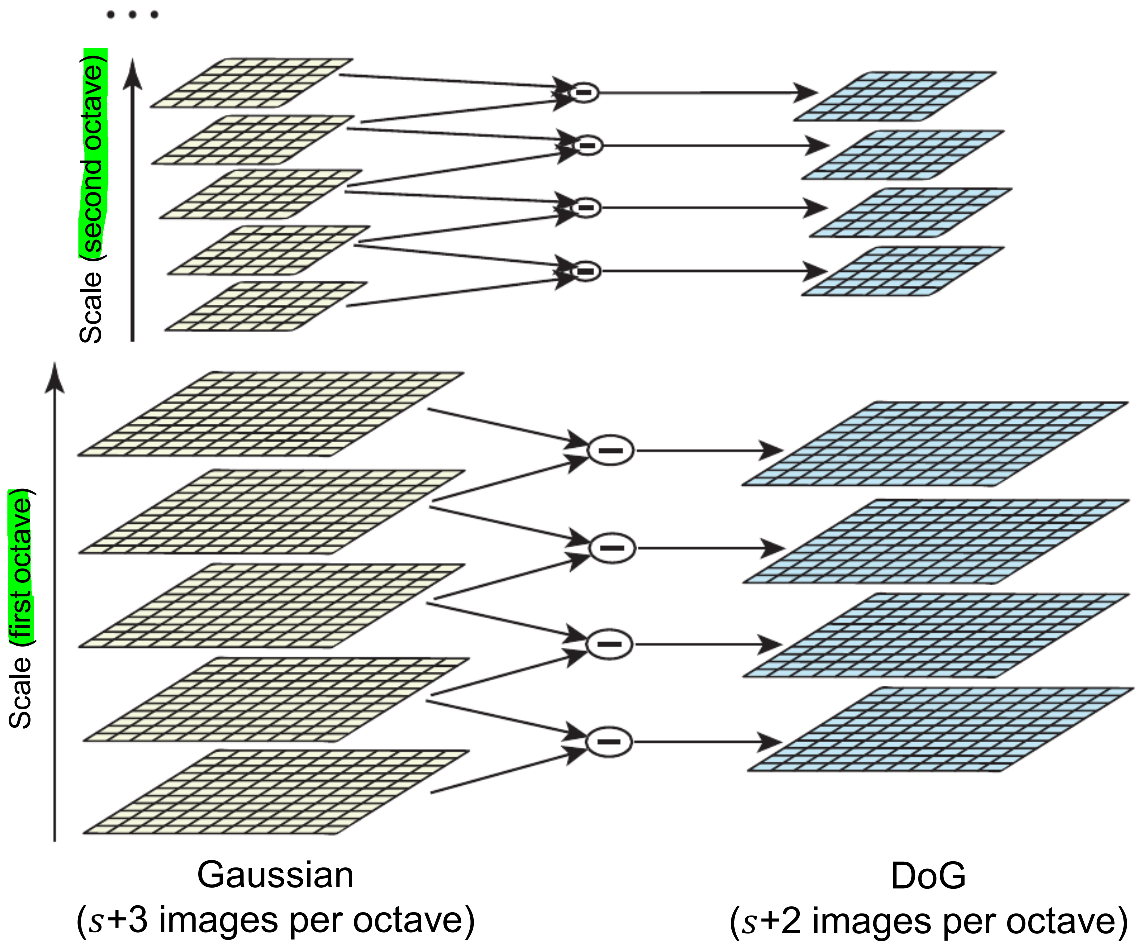
$\vdots$

$$\sigma_i = 2^i\sigma_0 = k^i\sigma_0$$

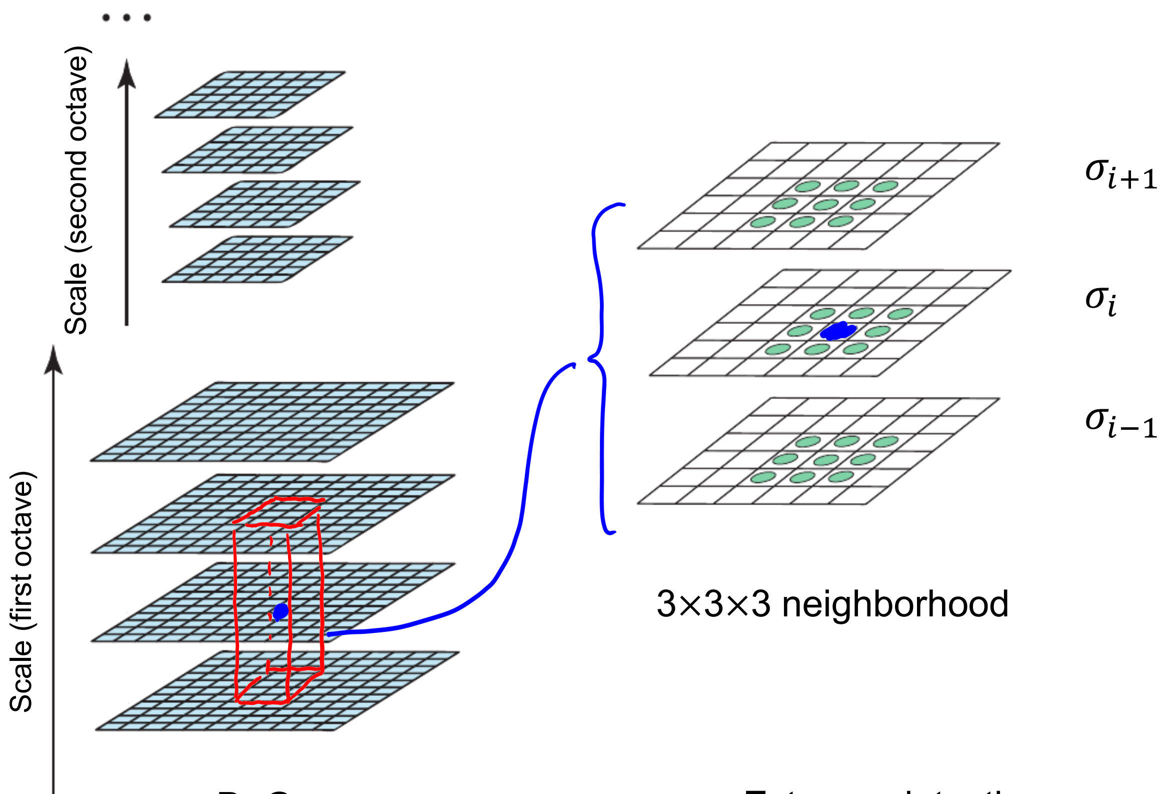
$\vdots$

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

σ_0 - original image



Scale Space Extrema Detection



DoG
(s+2 images per octave)

Extrema detection
(s maps per octave)

Scale space extremum detected if $\underbrace{\text{larger (smaller)}}_{\text{ignore sign}}$ than any of

2) Keypoint Localization

We obtain a set of extrema points $\mathrm{DoG}[x, y, \sigma]$

- For each extremum: position, scale, magnitude

Position and magnitude are refined by Taylor expansion

- An extremum may be located between two pixels

sub-pixel accuracy

Discard weak extrema by thresholding ($|\mathrm{DoG}[x, y, \sigma] < \theta|$)

- Noise, unstable features with low contrast, etc.

get rid of many keypoints

Keypoint Localization

Eliminating edge responses (keep only corners)

- DoG responds not only to keypoints but also to edges

Procedure analogous to Harris corner detector (cf. 6-46, Harris detector)

- Based on Hessian matrix

$$\boldsymbol{H} = \begin{bmatrix} \frac{\partial^2 s[x,y]}{\partial x^2} & \frac{\partial^2 s[x,y]}{\partial x \partial y} \\ \frac{\partial^2 s[x,y]}{\partial y \partial x} & \frac{\partial^2 s[x,y]}{\partial y^2} \end{bmatrix} = \begin{bmatrix} s_{xx}[x,y] & s_{xy}[x,y] \\ s_{yx}[x,y] & s_{yy}[x,y] \end{bmatrix}$$

- Edges exhibit highly disproportional eigenvalues of $\boldsymbol{H}$

$$\boxed{r = \frac{\lambda_1}{\lambda_2}}$$

$r \neq 1$ for edges

$r \approx 1$ for keypoints (because this is a corner)

Keypoint Localization

Eigenvalues are expensive to compute

- Use Harris approach instead

using $\frac{\lambda_1}{\lambda_2} = r$

$$\frac{\mathrm{Tr}(\boldsymbol{H})^2}{\det(\boldsymbol{H})} = \frac{(\lambda_1 + \lambda_2)^2}{\lambda_1 \lambda_2} = \frac{(r+1)^2}{r} \approx r \quad (\text{if } r \gg 1)$$

Discard extrema that are edges, i.e. whose Hessian obeys

$$\boxed{\frac{\mathrm{Tr}(\boldsymbol{H})^2}{\det(\boldsymbol{H})} \geq \frac{(r+1)^2}{r}}$$

$\Rightarrow$ only keep corners

Typical value for r is 10

3) Orientation Assignment

Keypoints are assigned **reference** orientation

- Keypoint descriptor relative to this orientation → **invariance to rotation**

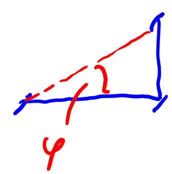
For each keypoint at (x, y, σ)

- Get gradient magnitude and orientation within a region around it

$$m[x,y] = \sqrt{(s[x+1,y,\sigma] - s[x-1,y,\sigma])^2 + (s[x,y+1,\sigma] - s[x,y-1,\sigma])^2}$$

DoG filtered image

$$\varphi[x,y] = \operatorname{atan} \frac{s[x,y+1,\sigma] - s[x,y-1,\sigma]}{s[x+1,y,\sigma] - s[x-1,y,\sigma]}$$



-2,-2	-1,-2	0,-2	1,-2	2,-2
-2,-1	-1,-1	0,-1	1,-1	2,-1
-2,0	-1,0	x, y	1,0	2,0
-2,1	-1,1	0,1	1,1	2,1
-2,2	-1,2	0,2	1,2	2,2

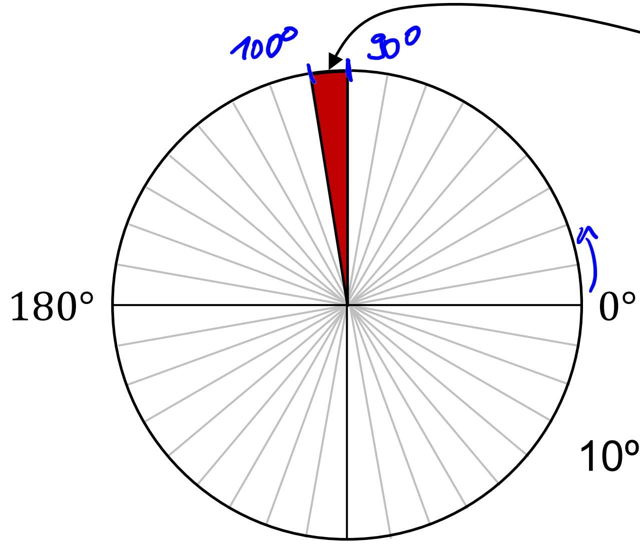
With this information orientation histogram is formed

calculate orientation at 5x5 points

Orientation Assignment

Orientation assigned by voting

5 votes



10° per sector

Orientation histogram

121°	122°	128°	131°	121°
120°	120°	124°	120°	123°
98°	90°	95°	102°	110°
97°	95°	101°	101°	70°
84°	79°	80°	80°	76°

sample 5×5 neighborhood

Votes proportional to:

- gradient magnitude $m[\cdot]$
- distance from keypoint (Gaussian window with 1.5σ)

Orientation Assignment

Extract dominant directions from histogram

- Most voted direction (histogram peak)
- Any local peak within 80% of the highest peak (if any)

If more than one dominant direction detected → multiple keypoints created (same location and scale, different orientation)

4) Keypoint Descriptor

Goal: highly distinctive and highly invariant

Procedure

1) Compute **relative** orientation and magnitude in a 16×16 neighborhood around keypoint => in order to be rotationally invariant
2) Form weighted histograms (8 bins) for 4×4 regions (weighted by magnitude, Gaussian window and distance from bin center)

Illustration with 8×8 neighborhood:

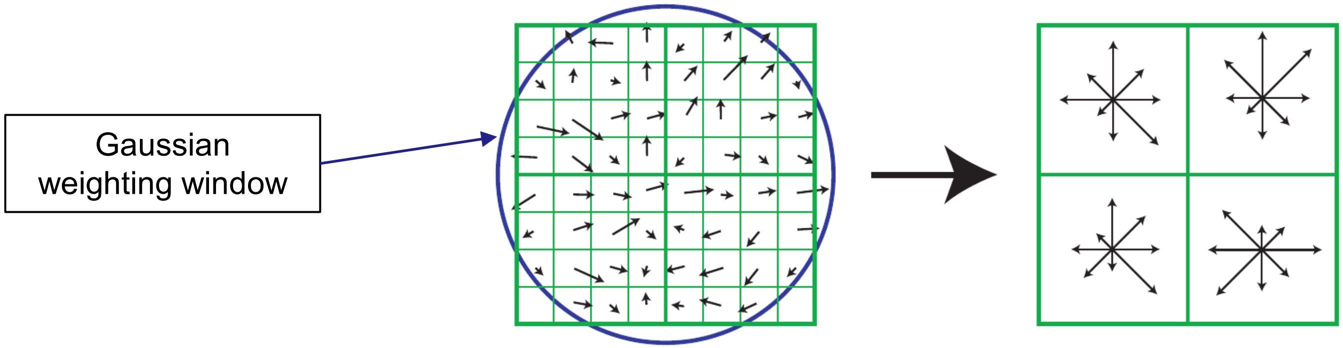


Image gradients at σ

Keypoint descriptor

Keypoint Descriptor

We obtain 16 histograms of 8 bins each

- Feature vector with 128 elements

Feature vector is normalized

- Reduces effects of illumination change

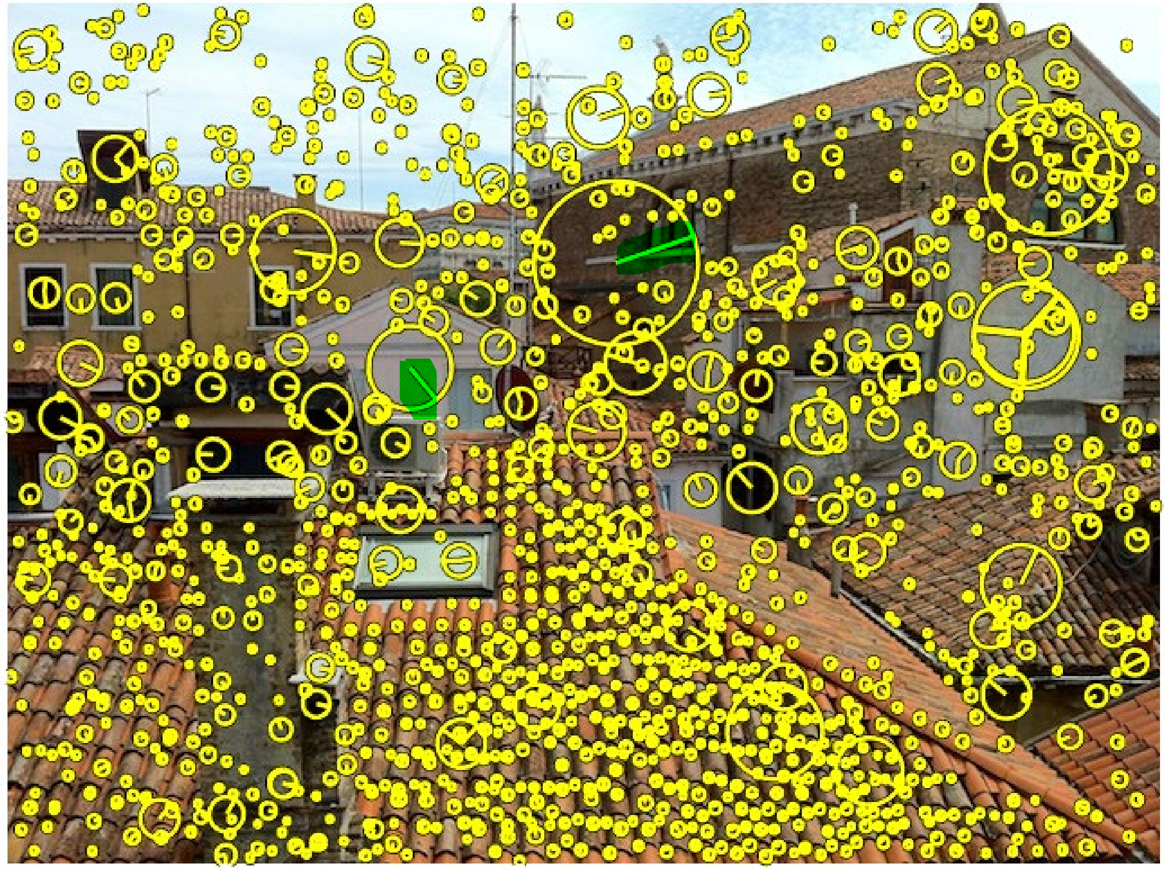
Normalized vector is up-clipped by 0.2 and renormalized

- Reduces the effect of changes in contrast
- Distribution of orientations is more important than magnitudes of large gradients
- **SIFT descriptor**

SIFT Example

SIFT Example

SIFT keypoints (scale + orientation)



Size of circle corresponds to scale at which blob was detected

SIFT Example

Descriptors of random 50 SIFT keypoints

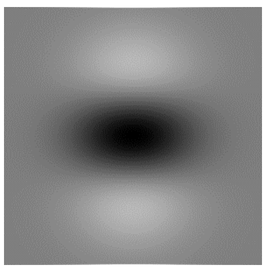
7.4 SURF

Speeded Up Robust Features (Herbert Bay, 2008)

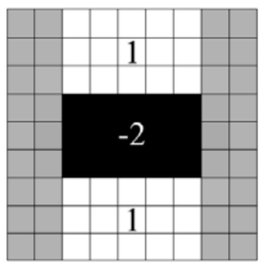
- Consists of detector and descriptor, inspired by SIFT
- Idea: simplify methods to essential

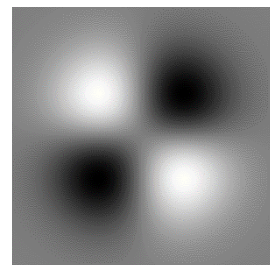
Detection based on Hessian matrix (as Harris, SIFT) using box filters

- Simple approximation of Gaussian second order partial derivatives

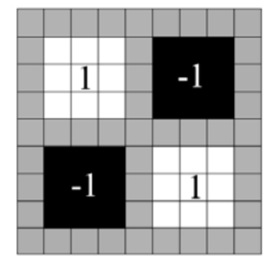


$\longrightarrow \dfrac{\partial^2 s[x,y]}{\partial y^2}$





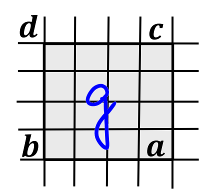
$\longrightarrow \dfrac{\partial^2 s[x,y]}{\partial x \partial y}$



Integral images s_Σ are used to calculate box filter output g

$$s_\Sigma[x,y] = \sum_{i=0}^{x} \sum_{j=0}^{y} s[i,j]$$

$$g = s_\Sigma[\boldsymbol{a}] - s_\Sigma[\boldsymbol{b}] - s_\Sigma[\boldsymbol{c}] + s_\Sigma[\boldsymbol{d}]$$



SURF

Blob detection by determinant of Hessian

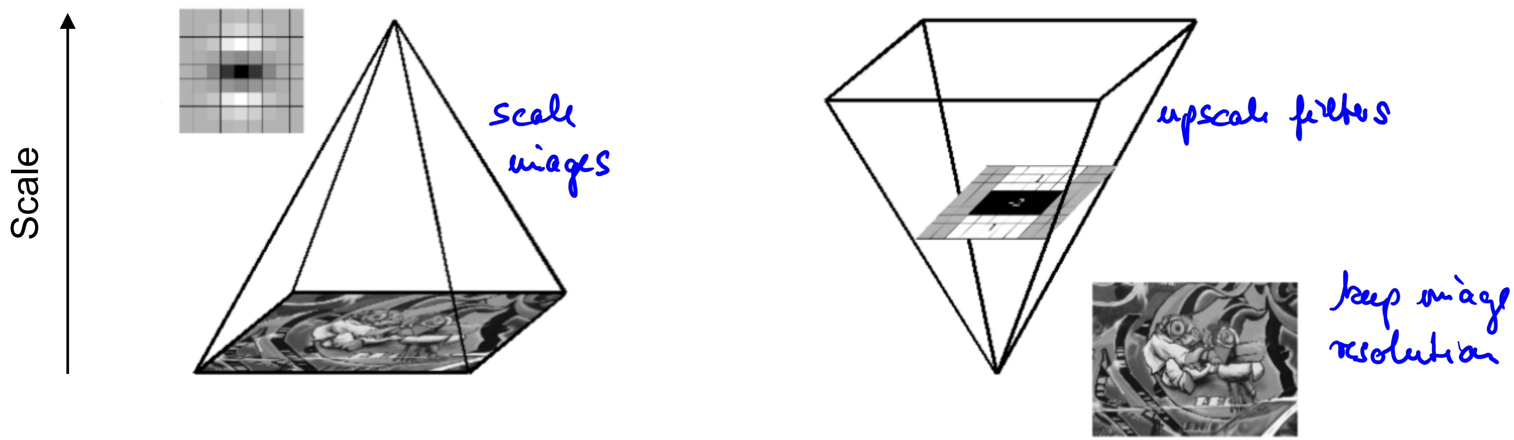
- Non-maximum suppression in 3×3×3 neighborhood

$$\det(\boldsymbol{H}_{\text{approx}}) \approx s_{xx}s_{yy} - (0.9s_{xy})^2$$

⇒ blob response
store in map over
different scales

Scale space is analyzed by up-scaling filter size

- Image size is fixed (unlike SIFT)

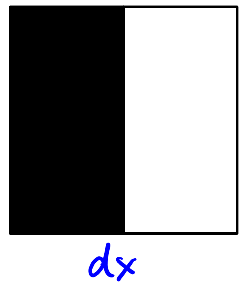


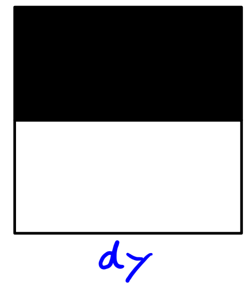
SIFT SURF

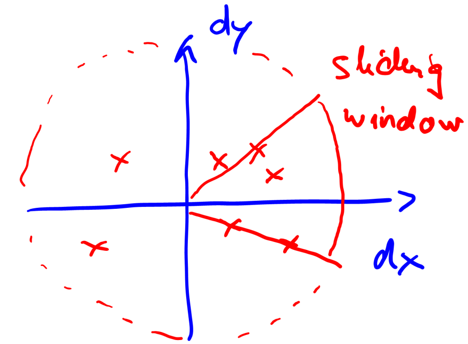
SURF Descriptor

1) Orientation assignment by voting procedure

- Haar wavelet responses in x and y directions



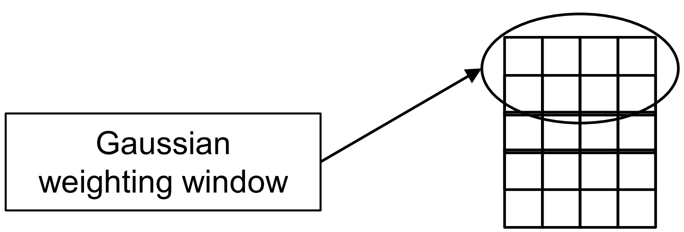


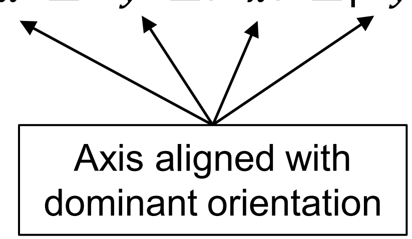


dominant orientation ≙ largest sum of dx, dy

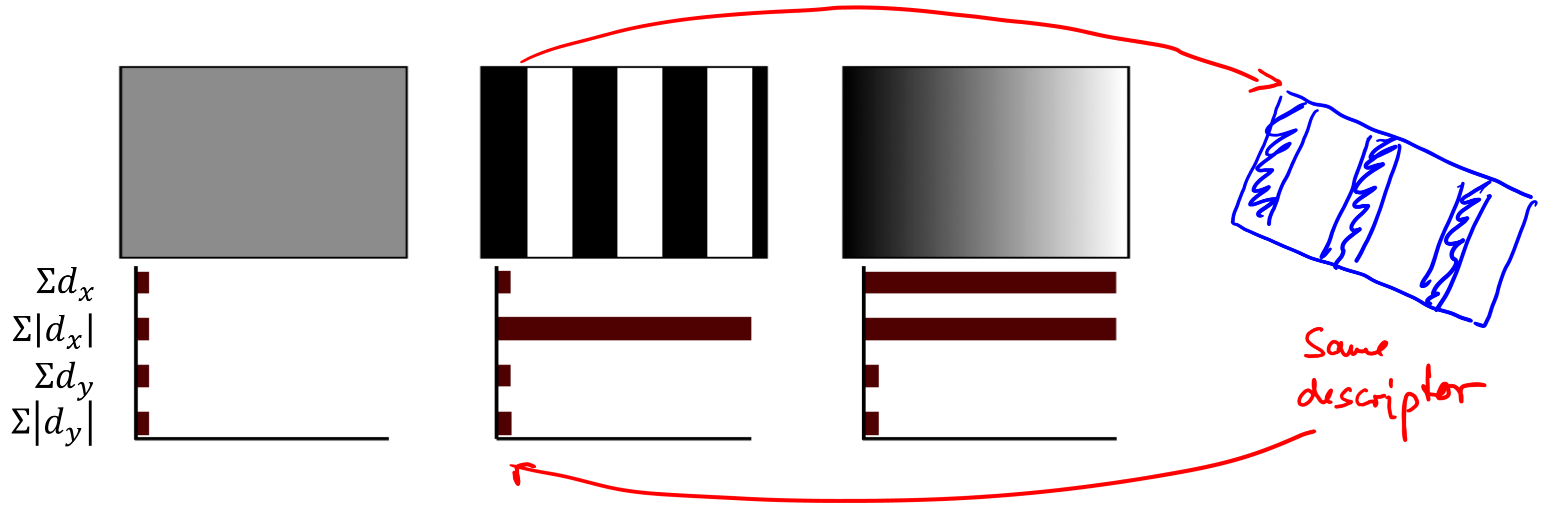
2) Region around keypoint is split up into 4×4 sub-regions

- Haar wavelet responses **with respect to** assigned orientation are computed for each sample (with Gaussian weighting as in SIFT)
- Descriptor vector for each sub-region $\boldsymbol{v} = \left(\sum d_x, \sum d_y, \sum|d_x|, \sum|d_y|\right)$





SURF Descriptor



SURF descriptor 4×4×4 = 64 dimensional vector

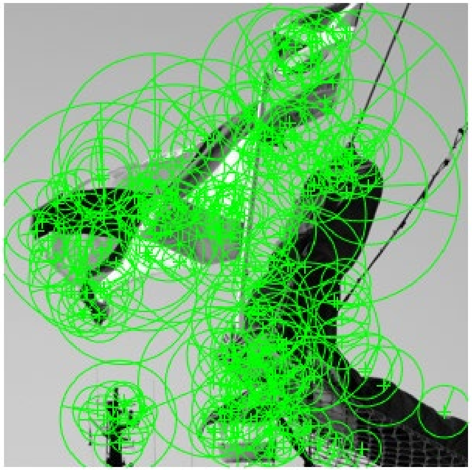
- Normalized for contrast invariance

Upright SURF (U-SURF)

- In many cases cameras do not rotate (much) so rotation invariance is an overkill
- Skips orientation assignment → faster to compute
- Robust to ±15º tilts

SURF Example

All SURF keypoints



10 strongest keypoints

Size of circle corresponds to scale at which blob was detected