Feature Detection / Description

Reference: Szeliski, Chapter 4

+ additional references on slides

Today's lecture

- Feature detection (continued from last lecture)
- Scale-Invariant Feature Transform (SIFT)

Very good resource on features detectors/descriptors (ECCV 2012 tutorial):

A. Vedaldi, J. Martas, K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman:
https://sites.google.com/site/eccv12features/slides

Feature detection

continued …

We have seen that the "standard" Harris detector **lacks scale invariance!**

Question: What can we do to obtain scale invariance ?

- One example: extract keypoints at multiple scales [Brown et al., 2005]

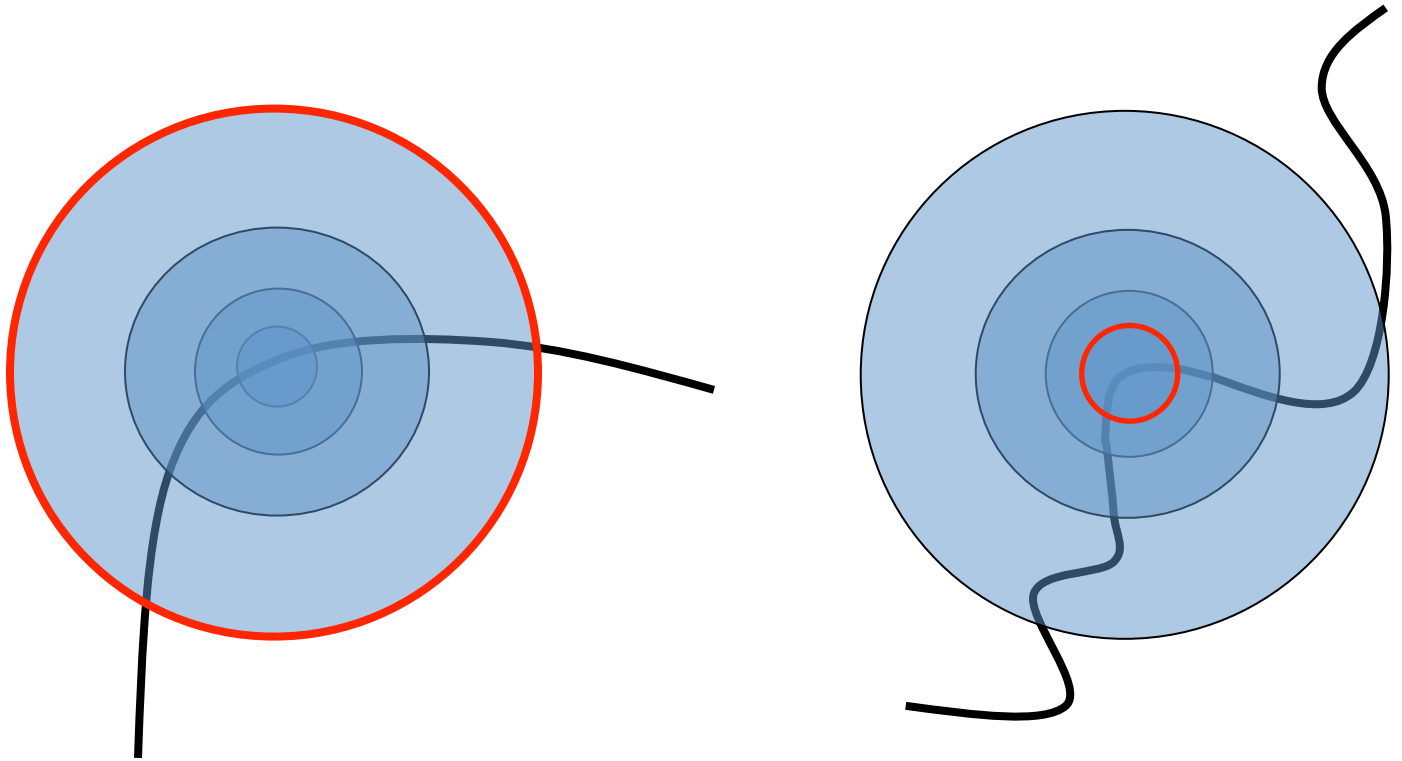
However, for matching, it is beneficial **not to match all features at all scales**, but to use features that are **stable** in both **scale AND location** [Lowe, 2004]

Lowe, D., "Distinctive Image Features from Scale-Invariant Keypoints", IJCV (60), pp. 91-101, 2004
Brown, M., Szeliski, R. and Winder, S., "Multi-image matching using multi-scale oriented patches, CVPR 2005

Feature detection

Scale invariance

Question: How do we choose corresponding circles **independently** in each image?

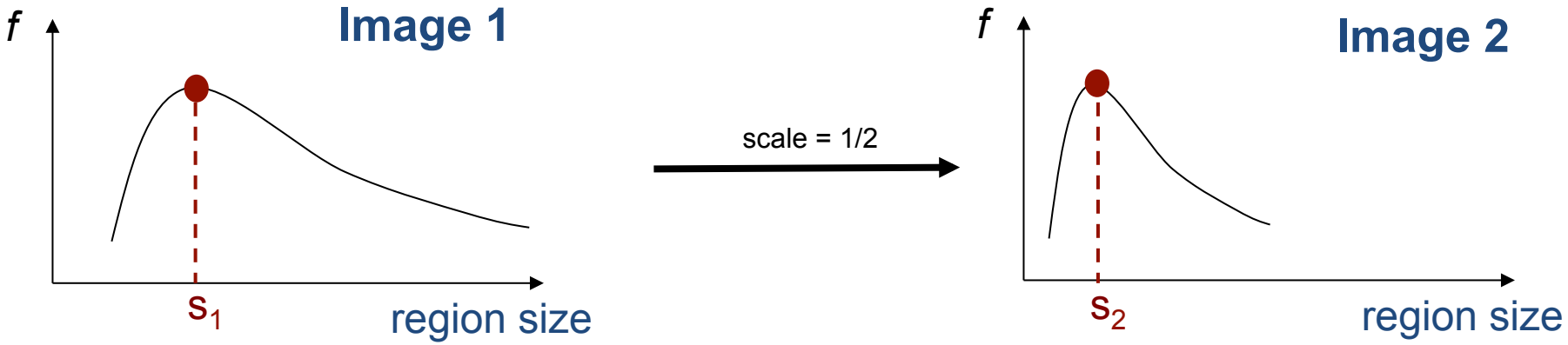


Slide adapted from A. Torralba

Feature detection

Scale invariance

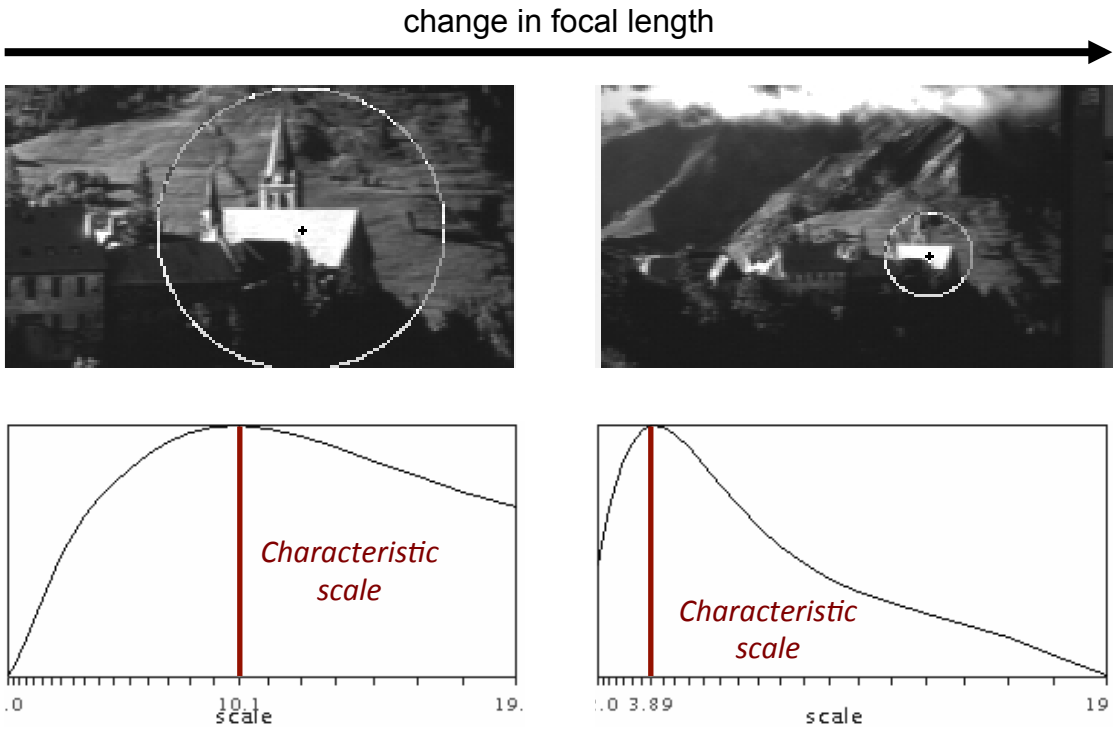
Solution: Design a function on the "region" which is - *by construction* - **scale invariant**! (*e.g.*, average intensity) and take a **local maximum** of that function!



Feature detection

Scale invariance

Example (from [Mikolajczyk, 2001])



Response function: "scale-normalized" Laplacian (performs well experimentally)

Mikolajczyk, K. and C. Schmid, "Indexing based on scale invariant interest points", ICCV, 2001

Feature detection

Scale-Invariant Feature Transform (SIFT) – **Part I**: Detector

Lowe's SIFT describes both a (1) keypoint **detector** and (2) keypoint **descriptor**!

The works of [Witkin, 1983; Lindeberg, 1994] on scale-space theory are essential to the development of SIFT.

Scale-space representation [Witkin, 1983]: Embedding of the original signal into a family of derived signals, constructed by convolution with a (one-parameter) family of Gaussian kernels.

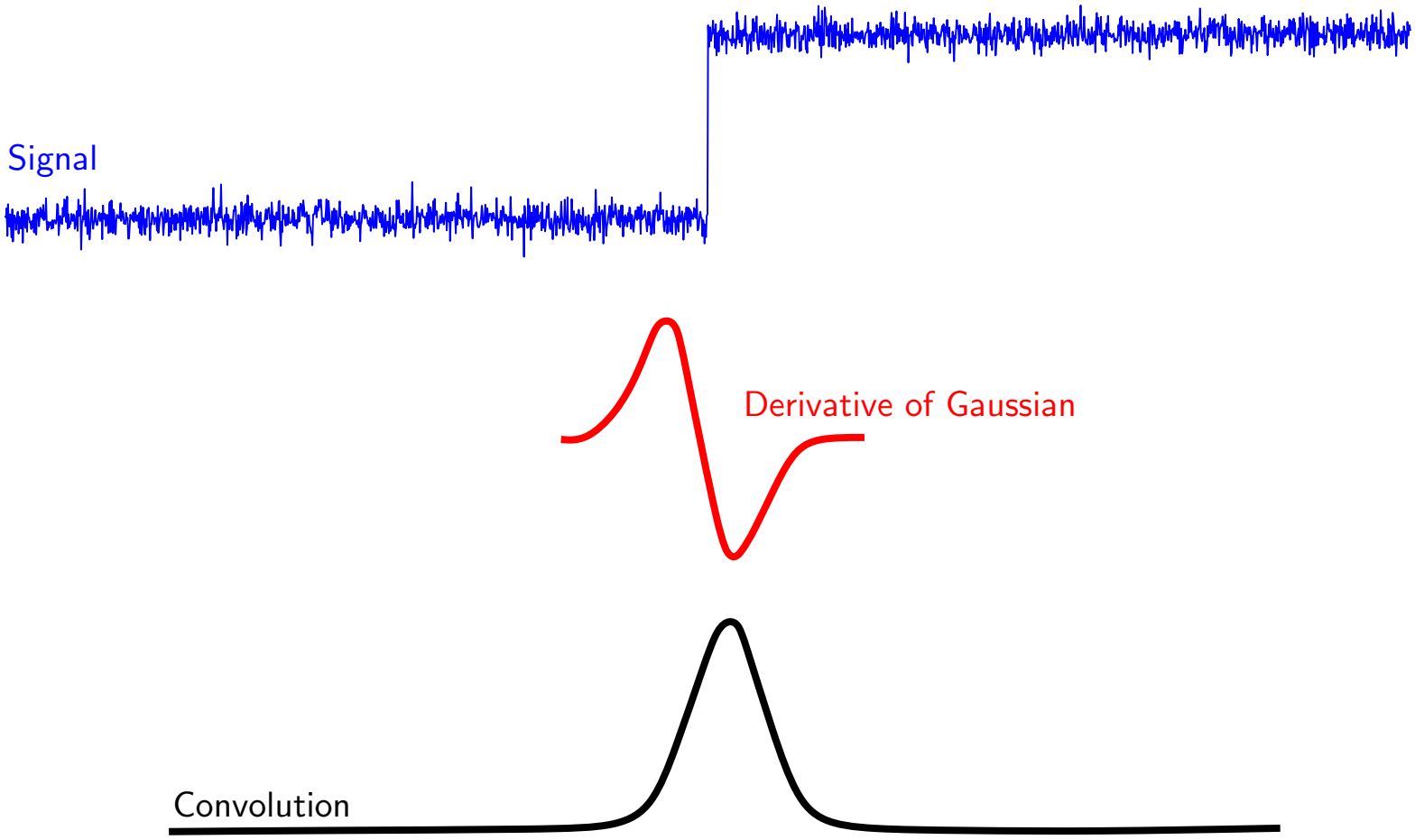
*The Gaussian kernel is the only reasonable kernel for true scale-space analysis [Lindeberg, 1994]**

*under a variety of assumptions, obviously (see paper)

Witkin, A., "Scale-space filtering", IJCAI, 1983
Lindeberg, T., "Scale-space theory: A basic tool for analyzing structures at different scales", J. Appl. Statistics, (2), pp. 224-270, 1994

Feature detection

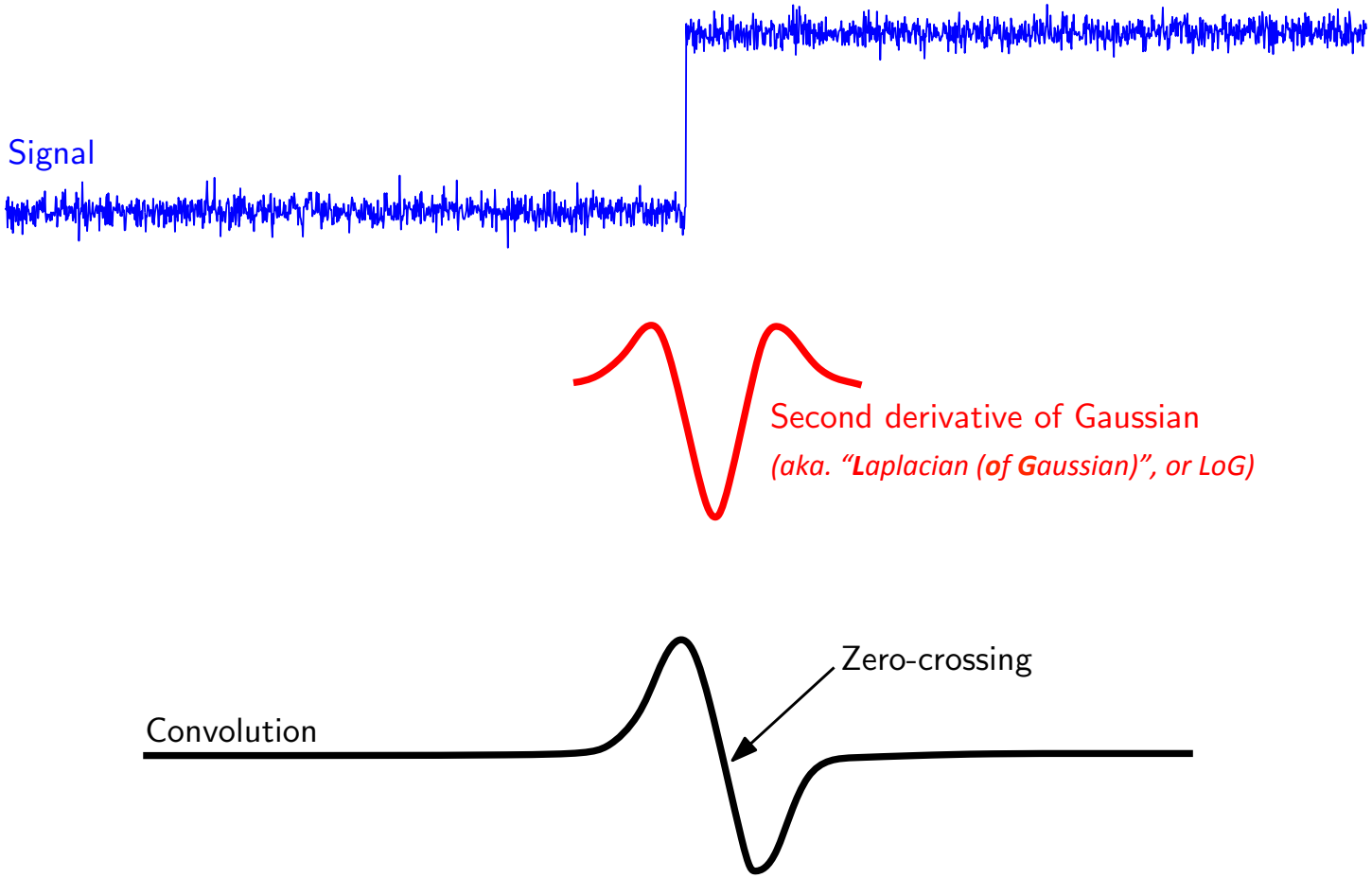
Quick recap of on edge detection



this figure only illustrates the principle

Feature detection

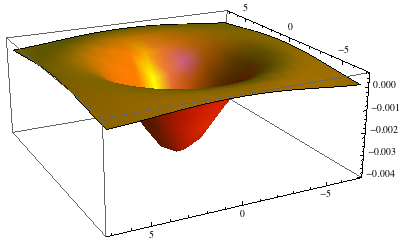
Quick recap of on edge detection



this figure only illustrates the principle

Feature detection

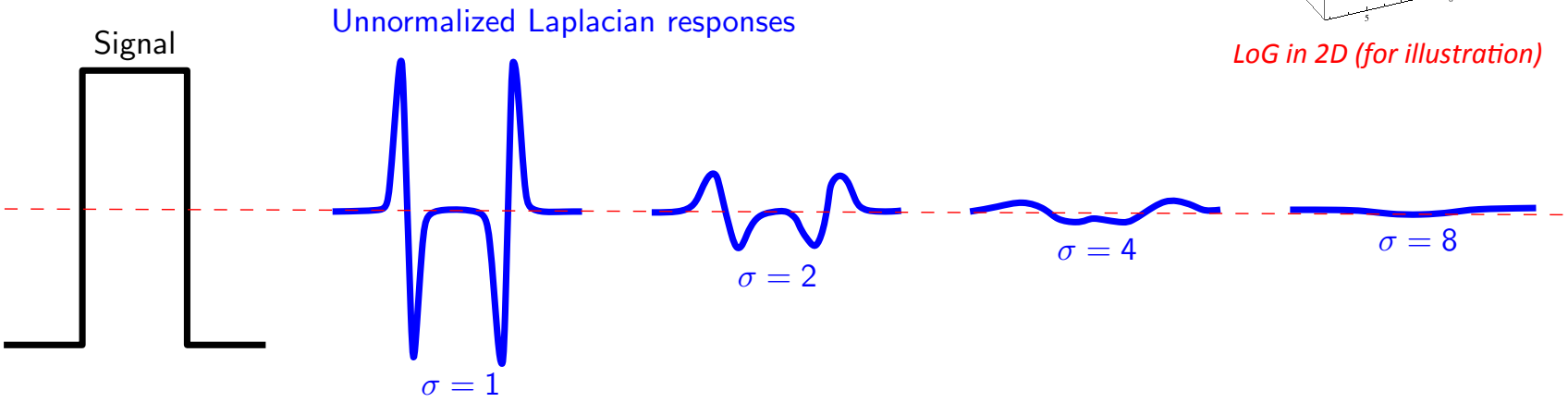
Why do we need scale normalization?



LoG in 2D (for illustration)

What is the problem?

- Laplacian responses decrease as scale increase

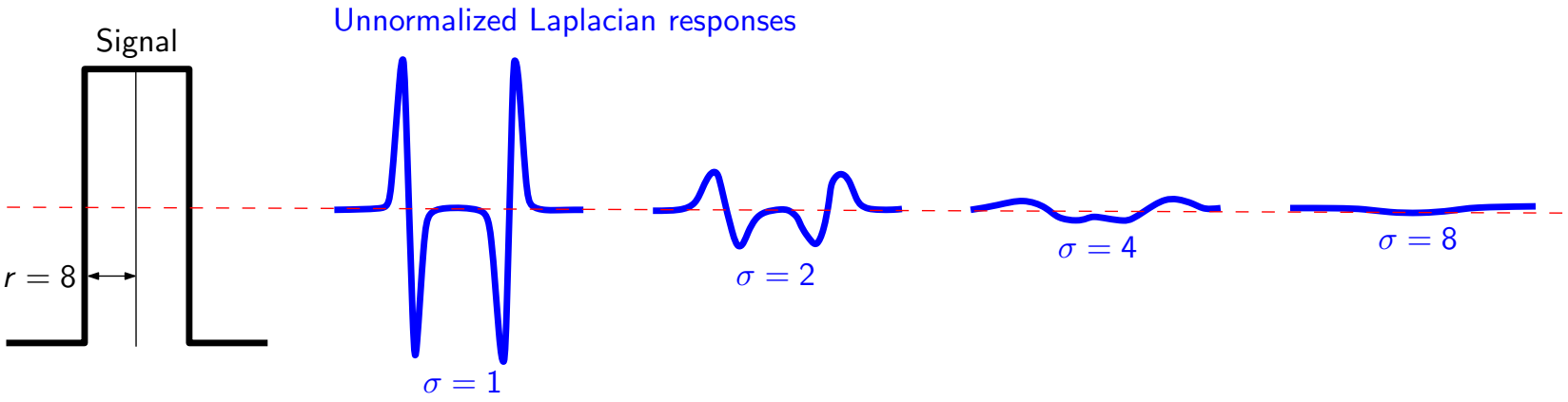


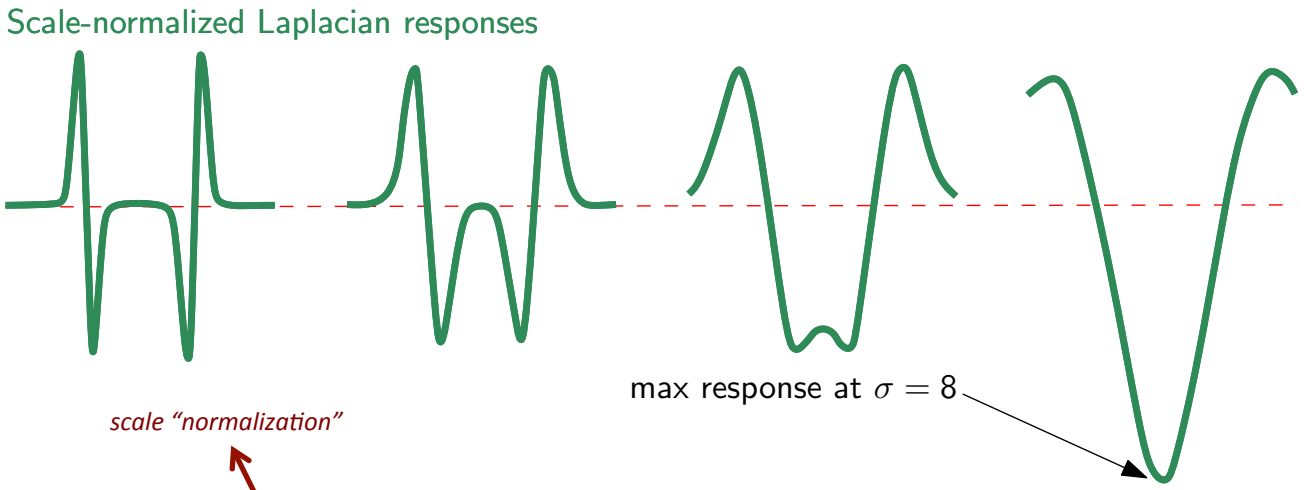
What can we do?

- pre-multiply by the scale → **scale normalization**

Feature detection

Why do we need scale normalization?





scale "normalization"

$$LoG(x, y, \sigma) = \sigma^2 \nabla^2 L(x, y, \sigma)$$

Image convolved with Gaussian (see next slide)

Feature detection

Scale-Invariant Feature Transform (SIFT) – **Part I**: Detector

We build the **scale-space representation** by

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

i.e., convolution of image with Gaussian of varying width

[Lowe, 1999, 2004] analyzes the **Difference-of-Gaussian (DoG)** function in scale-space:

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

Ok, but why? Because it is an approximation(**) to the "scale-normalized" **Laplacian-of-Gaussian (LoG)** from [Lindeberg, 1994]

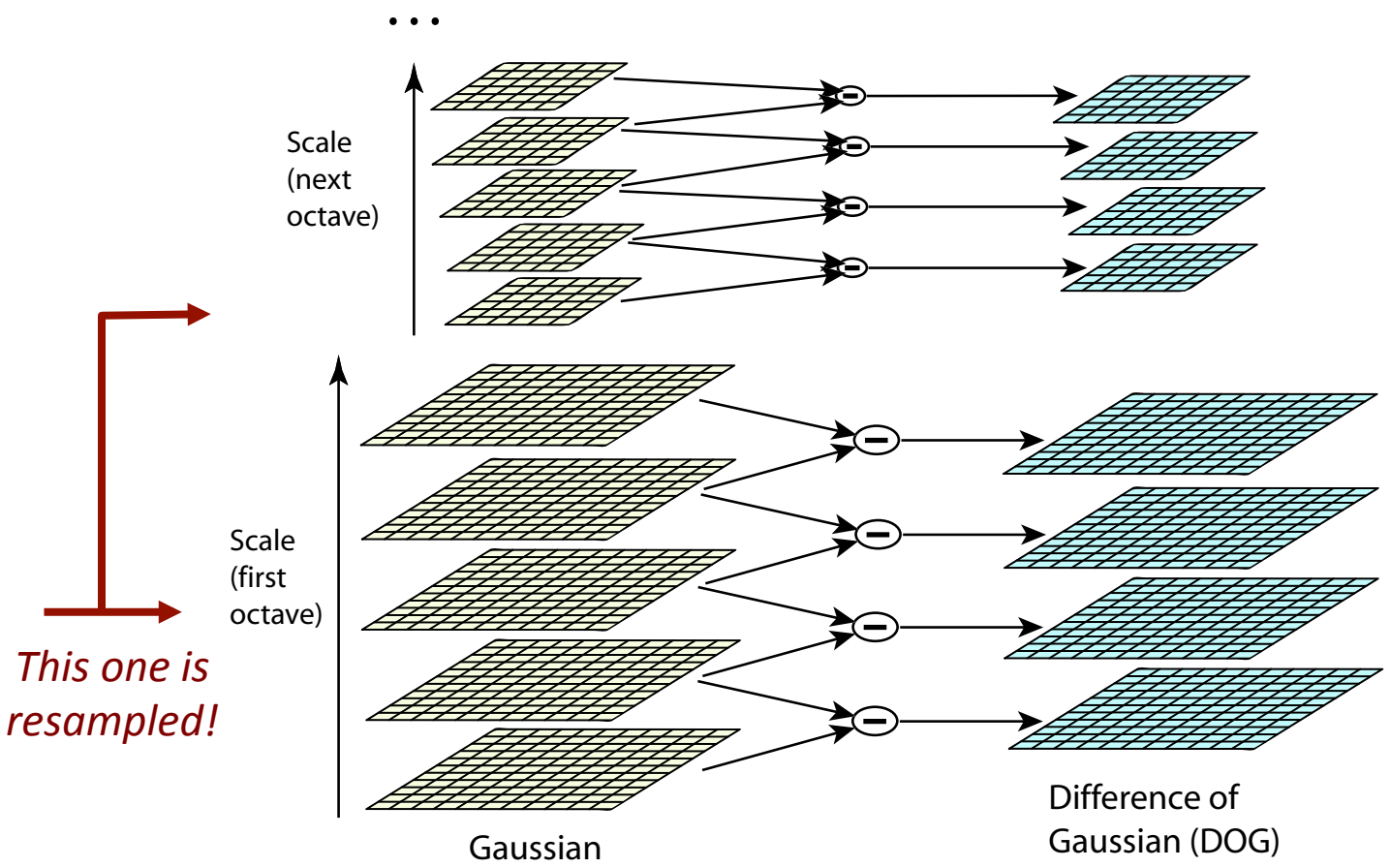
(**)*We'll do the derivation in class!*

Lindeberg, T., "Scale-space theory: A basic tool for analyzing structures at different scales", J. Appl. Statistics, (2), pp. 224-270, 1994
Lowe, D., "Object recognition from local scale-invariant features", ICCV, 1999
Lowe, D., "Distinctive Image Features from Scale-Invariant Keypoints", IJCV (60), pp. 91-101, 2004

Feature detection

Scale-Invariant Feature Transform (SIFT) – **Part I**: Detector

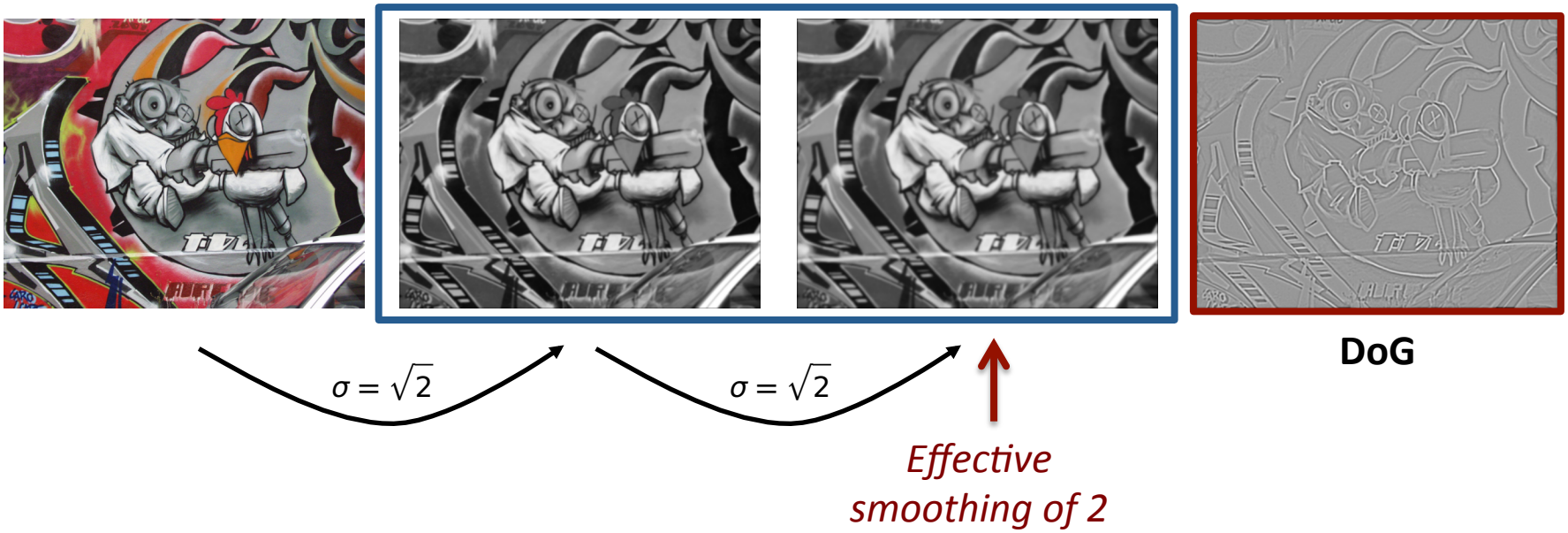
Typically, one octave is divided into **s** scales ($k = 2^{1/s}$)



Feature detection

Scale-Invariant Feature Transform (SIFT) – **Part I**: Detector

Example (to generate one DoG image)**:**



Feature detection

Scale-Invariant Feature Transform (SIFT) – **Part I**: Detector

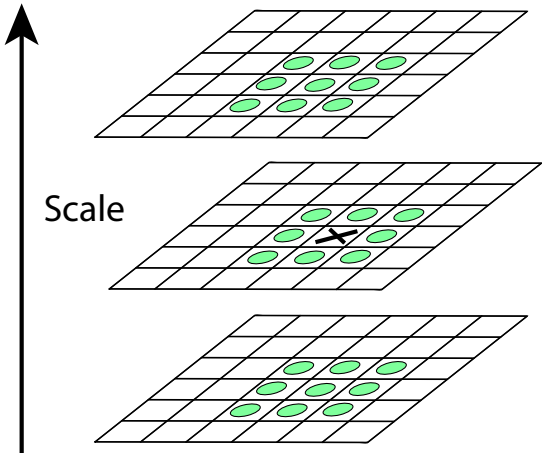
Computational steps for SIFT

- Local extrema detection
- Keypoint localization
- Orientation assignment
- Local descriptor computation

Feature detection

Scale-Invariant Feature Transform (SIFT) – **Part I**: Detector

Local extrema detection

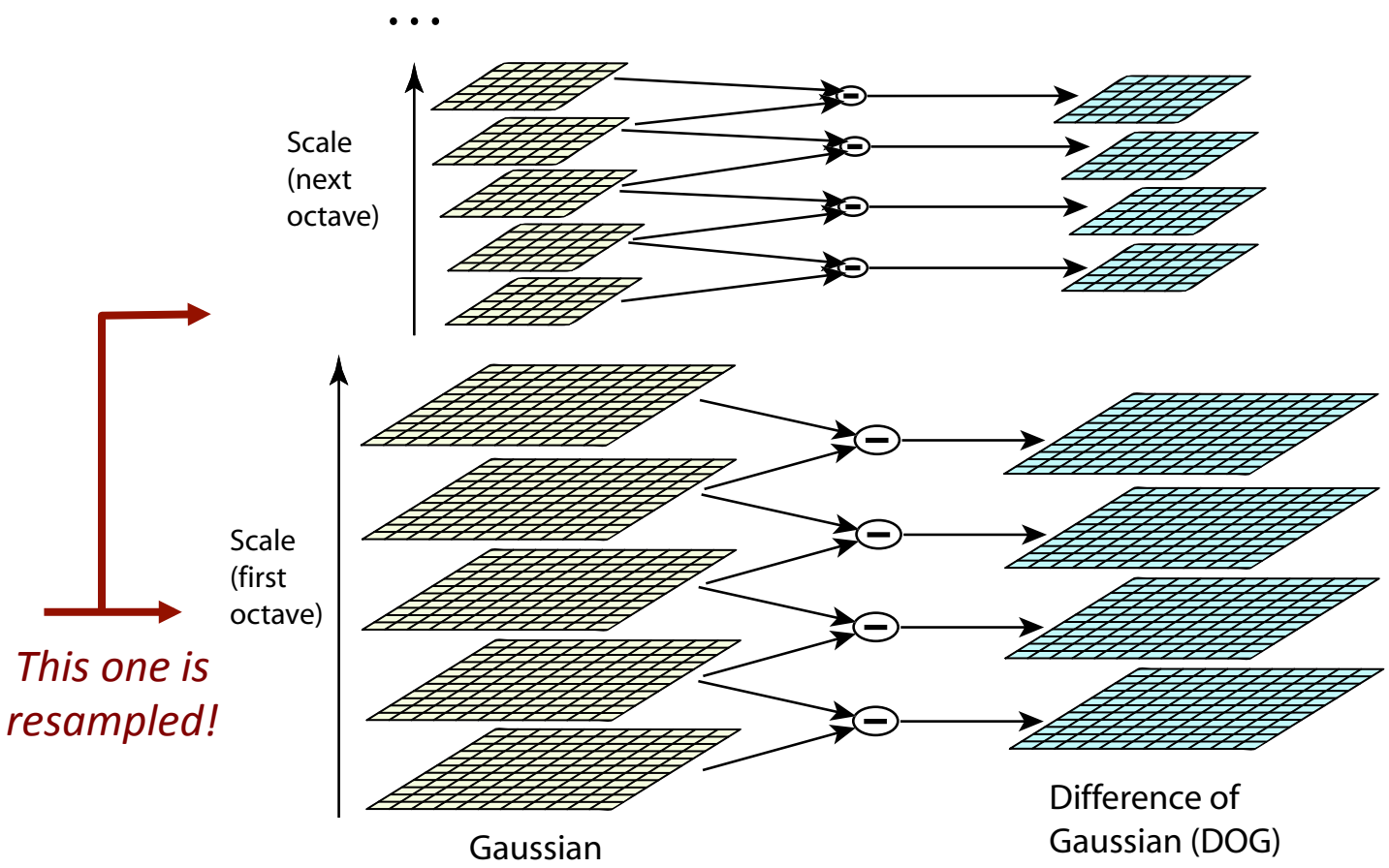


Strategy: Compare each sample to its 8 local neighbors and the 9 neighbors in the scale above and below!

Feature detection

Scale-Invariant Feature Transform (SIFT) – **Part I**: Detector

Now, our previous picture becomes more obvious: E.g., for s=2, we need s+3 **smoothed** images, so that local extrema detection covers **one** octave!



Feature detection

Scale-Invariant Feature Transform (SIFT) – **Part I**: Detector

Keypoint localization

After candidate detection, fit a **3D quadratic function** to local sample points
→ then, determine (interpolated) location of the maximum!

[Brown & Lowe, 2002] propose a Taylor series expansion of D around **each** candidate keypoint (here, **x** is the offset!).

$$D(\mathbf{x}) = D + \frac{\partial D^{\mathsf{T}}}{\partial \mathbf{x}}\mathbf{x} + 0.5\mathbf{x}^{\mathsf{T}}\frac{\partial^2 D^{\mathsf{T}}}{\partial \mathbf{x}^2}\mathbf{x}$$

Then, take the derivative with respect to **x** and set it to **zero**; We get

$$\hat{\mathbf{x}} = -\frac{\partial^2 D^{-1}}{\partial \mathbf{x}^2}\frac{\partial D}{\partial \mathbf{x}}$$

Feature detection

Scale-Invariant Feature Transform (SIFT) – **Part I**: Detector

Solve the resulting 3x3 linear system:

$$\begin{pmatrix} \frac{\partial^2 D}{\partial \sigma^2} & \frac{\partial^2 D}{\partial \sigma y} & \frac{\partial^2 D}{\partial \sigma x} \\ \frac{\partial^2 D}{\partial \sigma y} & \frac{\partial^2 D}{\partial y^2} & \frac{\partial^2 D}{\partial y x} \\ \frac{\partial^2 D}{\partial \sigma x} & \frac{\partial^2 D}{\partial y x} & \frac{\partial^2 D}{\partial x^2} \end{pmatrix} \begin{pmatrix} \sigma \\ y \\ x \end{pmatrix} = - \begin{pmatrix} \frac{\partial D}{\partial \sigma} \\ \frac{\partial D}{\partial y} \\ \frac{\partial D}{\partial x} \end{pmatrix}$$

If the offset **x is > 0.5** (in any dimension), change the sample point and perform interpolation around this point (since extrema is closer to other point) → Eventually, the final offset is added to the original keypoint location **AND**, if

$$|D(\hat{\mathbf{x}})| = \left| D + 0.5 \frac{\partial D^\top}{\partial \mathbf{x}} \hat{\mathbf{x}} \right|$$

(obtained by using the solution to the offset in the Taylor expansion)

is **< 0.03** ([Lowe, 2004] assumes pixel in [0,1]), the **keypoint is discarded!**

Feature detection

Scale-Invariant Feature Transform (SIFT) – **Part I**: Detector

Problem: DoG can have strong response along edges!

$$\mathbf{H} = \begin{pmatrix} D_{xx} & D_{xy} \\ D_{yx} & D_{yy} \end{pmatrix}$$

This Hessian is computed using **finite-difference filters**. Similar to the Harris detector, a **response function** is then defined:

$$\frac{\mathrm{tr}(\mathbf{H})^2}{\det(\mathbf{H})} = \frac{(\lambda_{\max} + \lambda_{min})^2}{\lambda_{max}\lambda_{\min}} = \frac{(r+1)^2}{r}, \text{ with } \lambda_{\max} = r\lambda_{\min}$$

This function only depends on the eigenvalue ratio. We **reject keypoints** if

$$\frac{\mathrm{tr}(\mathbf{H})^2}{\det(\mathbf{H})} = \frac{(\lambda_{\max} + \lambda_{min})^2}{\lambda_{max}\lambda_{\min}} < \frac{(1+r)^2}{r}$$

Feature detection

Scale-Invariant Feature Transform (SIFT) – **Part I**: Detector

Remember: First eigenvector of Hessian is the direction of largest curvature → along edge we have a large first EV & small second EV! *Typical choice: r=10!*
(Note: we have a minimum at r=1)

Orientation assignment:

Key idea is to compute the SIFT descriptor relative to the dominant orientation at the keypoint!

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

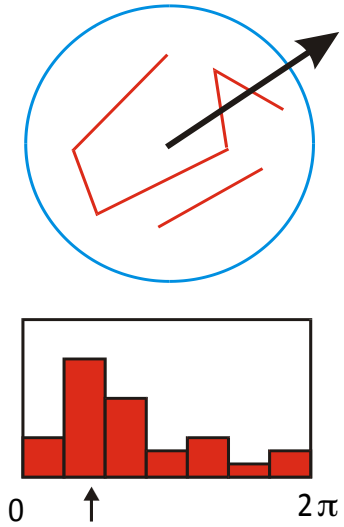
$$\theta(x,y) = \tan^{-1}\left(\frac{L(x,y+1) - L(x,y-1)}{L(x+1,y) - L(x-1,y)}\right)$$

***Note**: (1) L is the Gaussian-smoothed image! (2) the scale "closest to keypoint" is selected to fix L (needed, since we fit the quadratic function during refinement).*

Feature detection

Scale-Invariant Feature Transform (SIFT) – **Part I**: Detector

An orientation histogram (typically **36 bins**) is computed:



More details …

- Samples which are added to the histogram are weighted by gradient magnitude **AND** a Gaussian-weighted circular window with width = 1.5 times the scale of the keypoint!
- Bins within 80% of maximum → **new keypoint** (same position + scale, but different orientation)
- **Parabola is fit to the 3 histogram values closest to peak** → interpolate peak position

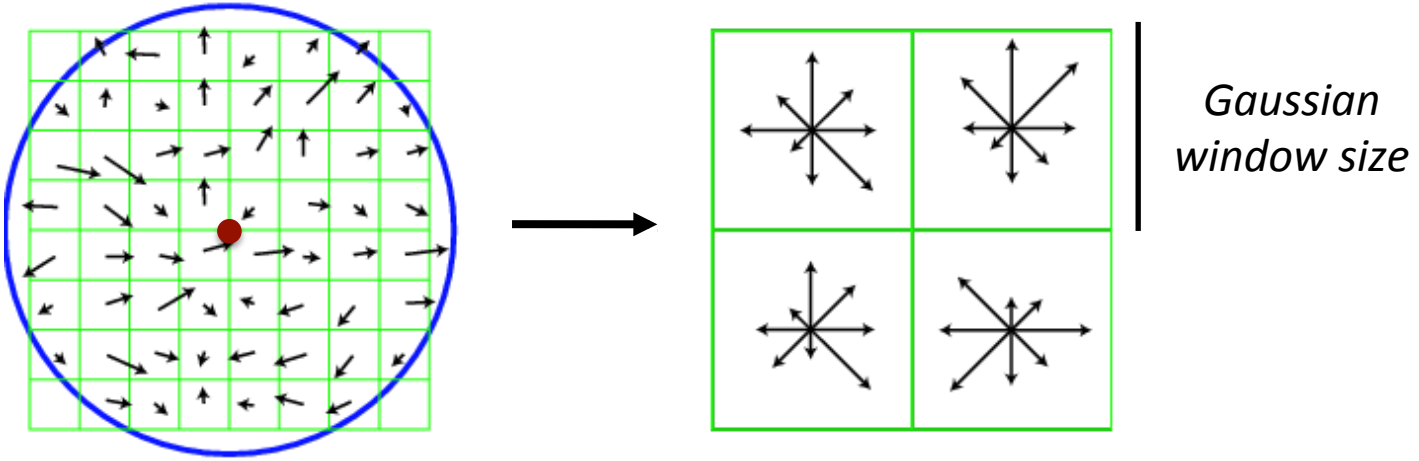
Feature description

Scale-Invariant Feature Transform (SIFT) – **Part II**: Descriptor

Local descriptor computation *(i.e.*, the "SIFT descriptor"):

So far, we have a repeatable (stable) 2D coordinate system (through keypoints)

The descriptor is motivated by work of [Edelman et al, 1997] in the field of biological vision (they looked at complex neurons in the primary visual cortex).



***Key idea**: the location of the gradient on the "retina" is allowed to shift (over a small receptive field), as opposed to being locally fixed.*

Edelman S., Intrator, N. and T. Poggio, "Complex Cells and Object Recognition", Unpublished Manuscript, 1997

Feature description

Scale-Invariant Feature Transform (SIFT) – **Part II**: Descriptor

1. Gradient sampling

The coordinates of the descriptor and the orientations are **rotated relative to the determined keypoint orientation**.

2. Gaussian smoothing

A Gaussian with a width one half of the descriptor window is used to smooth the gradient magnitudes (see figure on previous slide).

3. Orientation histogram

Essentially, we build a 3-D histogram with (N x N x M bins); *Typical choice: 4x4x8 (4x4 spatial bins, 8 orientation bins) → 128-dimensional descriptor!*

4. Histogram normalization / clamping

l_2 normalization (→ unit length) + clamping at 0.2 (then re-normalize).

Feature description

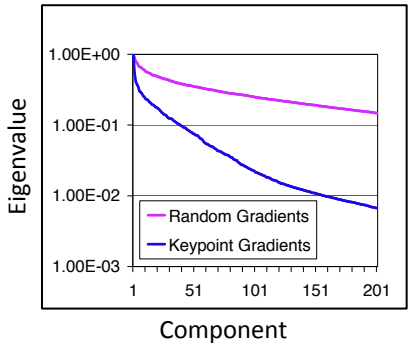
Scale-Invariant Feature Transform (SIFT) – Extensions & Advances

PCA-SIFT [Ke & Sukthankar, 2004]:

- Keypoint detection remains the same; **descriptor is different**
- Compute (x,y) gradients over a **41x41 patch** (centered on keypoint and oriented relative to the dominant keypoint orientation)
- l_2-normalize the resulting 3042-dimensional (39*39*2) vector
- Dimensionality reduction, via PCA, to 36 dimensions!

The authors argue that the remaining variation (after factoring out orientation & scale by keypoint detection) can be captured via PCA!

Implementation online:
http://www.cs.cmu.edu/~yke/pcasift/

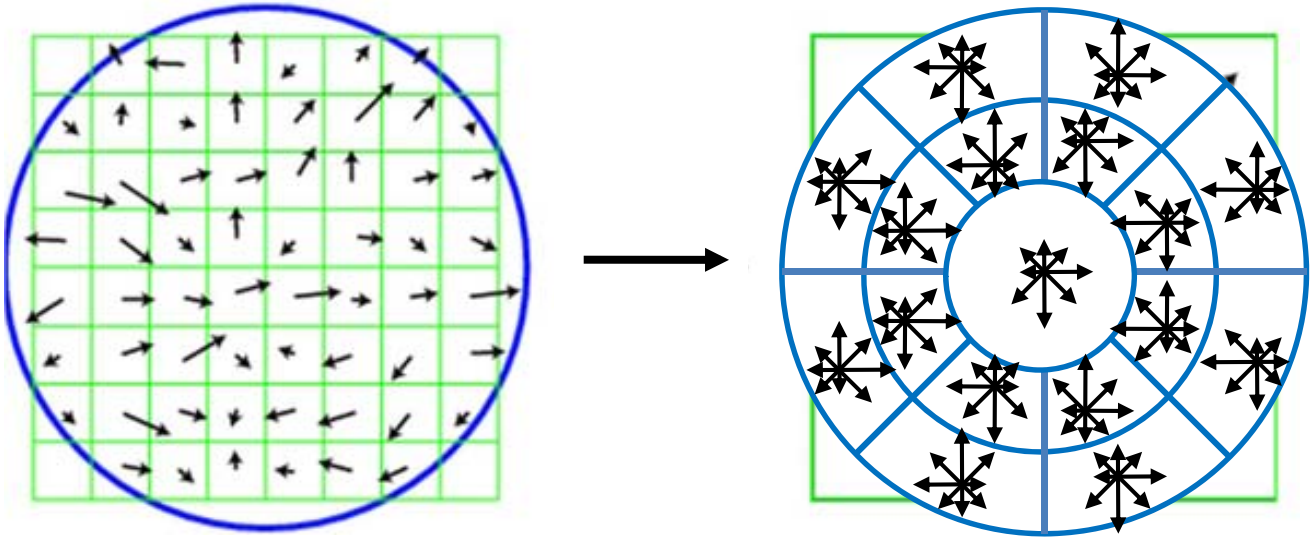


Feature description

Scale-Invariant Feature Transform (SIFT) – Extensions & Advances

GLOH [Mikolajczyk, 2005]:

- Abbreviation for "Gradient Location Orientation Histogram"
- Log-polar binning, instead of 4x4 binning as in "standard" SIFT



Mikolajczyk K. and C. Schmid, "A performance evaluation of local descriptors, PAMI 27 (10), pp. 1615–1630, 2005

Discussion

What do you think can we do with all those local descriptors ?

(apart from the obvious matching/stitching applications)

References

Best reference: read D. Lowe's IJCV paper, available from

https://www.robots.ox.ac.uk/~vgg/research/affine/det_eval_files/lowe_ijcv2004.pdf