Computer Vision

6. Texture, Local Feature Points and Descriptors

I-Chen Lin

College of Computer Science
National Yang Ming Chiao Tung University

Objective

- Key issue: How do we find or match important features in images?
- Topics:
  - Texture and image features.
  - Pyramid of steerable filters.
  - SIFT

Some contents are from the reference lecture notes or project pages:

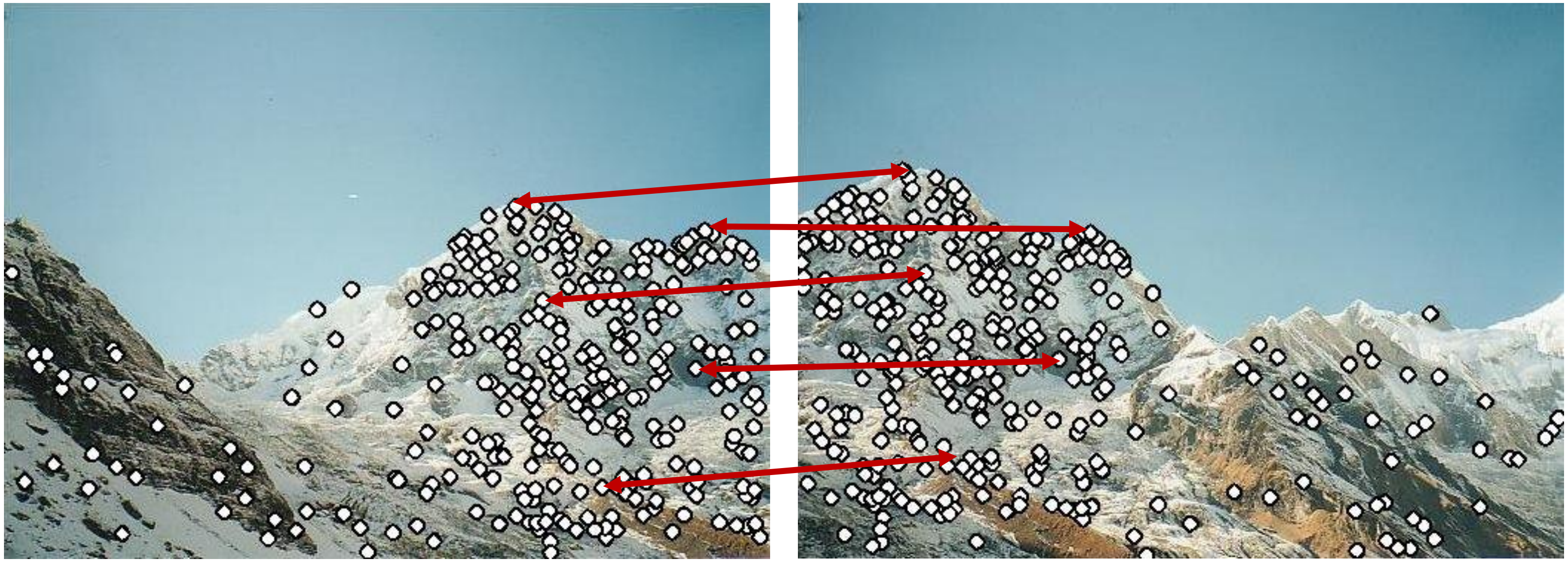
- D. Lowe, Lecture note "Distinctive Image Features from Scale-Invariant Keypoints ", UBC, CA.
- O. Pele, the presentation slides of "SIFT: Scale Invariant Feature Transform."
- Prof. D. Lowe, Computer Vision, UBC, CA.
- Prof. T. Darrell, Computer Vision and Applications, MIT.
- Prof. D.A. Forsyth, Computer Vision, UIUC.
- http://graphics.cs.cmu.edu/people/efros/research/EfrosLeung.html

Object instance recognition

Image matching from photo collection

N. Snavely et al., “Photo Tourism: Exploring Photo Collections in 3D,” Proc. ACM SIGGRAPH’06.

Tracking and correspondence

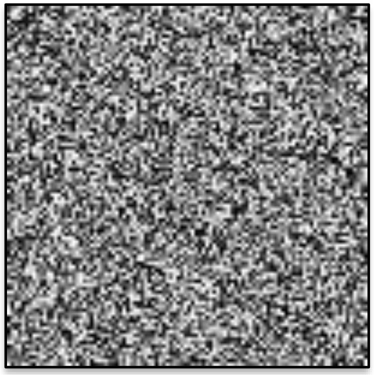


M.Brown, D.G. Lowe, “Recognising Panoramas,” Proc. ICCV’03.

Texture

- We have taught how to find the import points, such as corners.
- Our next step is to analyze the local appearances for matching, tracking and so on.
- How to capture the essence of texture?

Repeated/structured



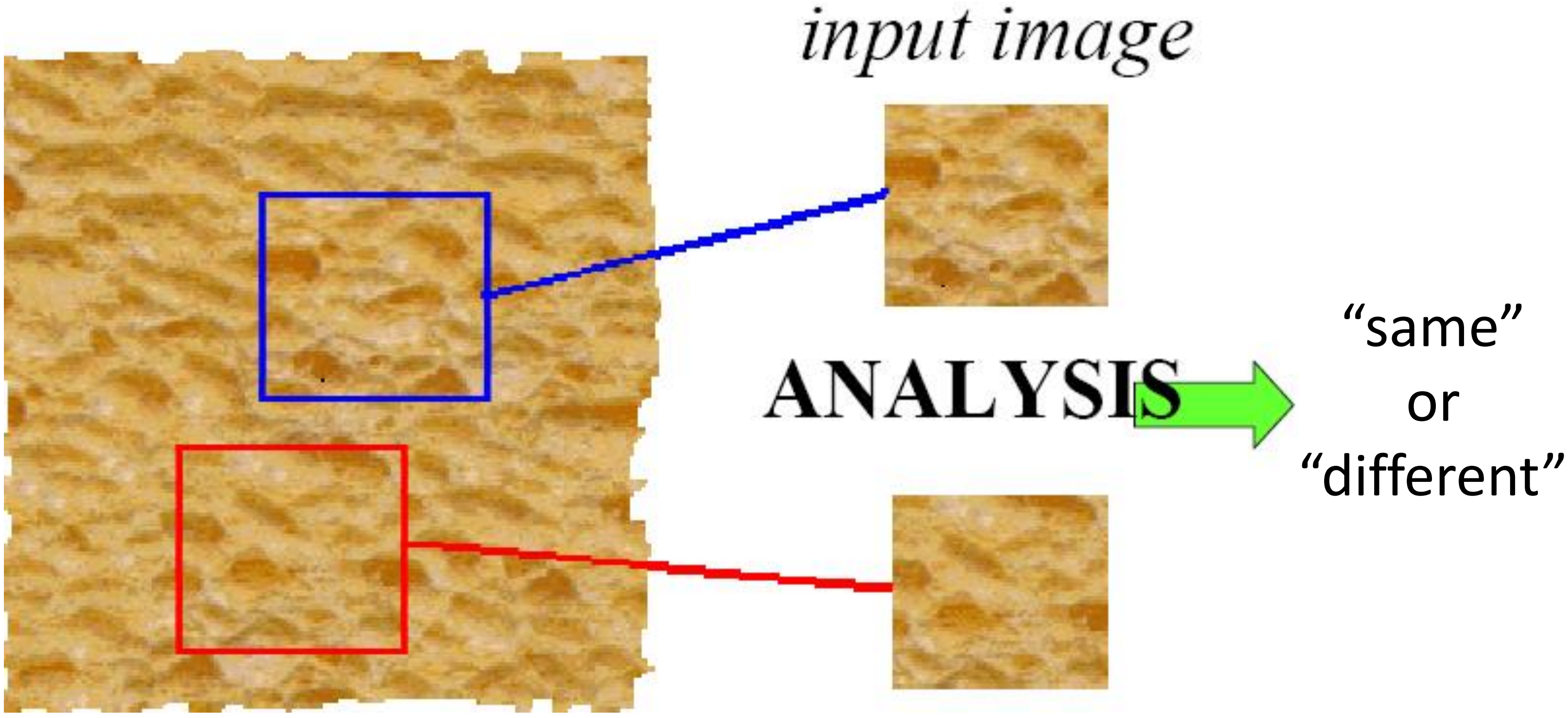
stochastic

Both?

http://graphics.cs.cmu.edu/people/efros/research/EfrosLeung.html

Texture analysis

- Compare textures and decide whether they are of the same "stuff".



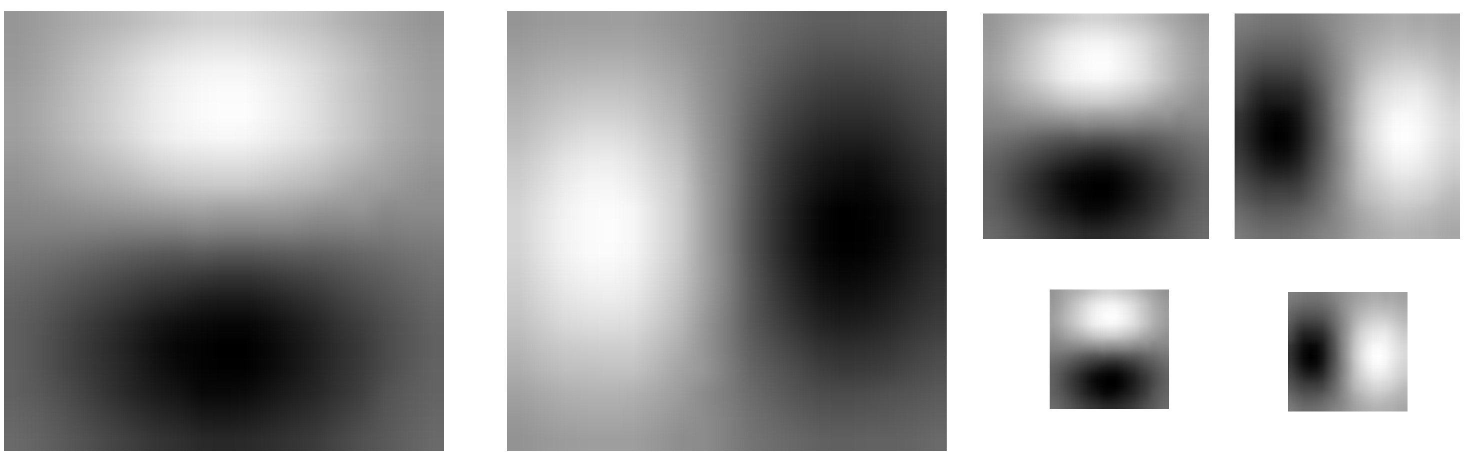
http://graphics.cs.cmu.edu/people/efros/research/EfrosLeung.html

Representing textures

- Observation
  - textures are made up of sub-elements, repeated over a region with similar statistical properties.
- Texture representation
  - Find the sub-elements, and represent their statistics ?!
- What filters can find the sub-elements?
  - Human vision suggests spots and oriented filters at a variety of different scales
- What statistics?
  - Mean of each filter response over region
  - Other statistics can also be useful

Derivative of Gaussian filters

- Derivatives of Gaussian filters measure magnitudes and direction of image gradients.



Texture representation

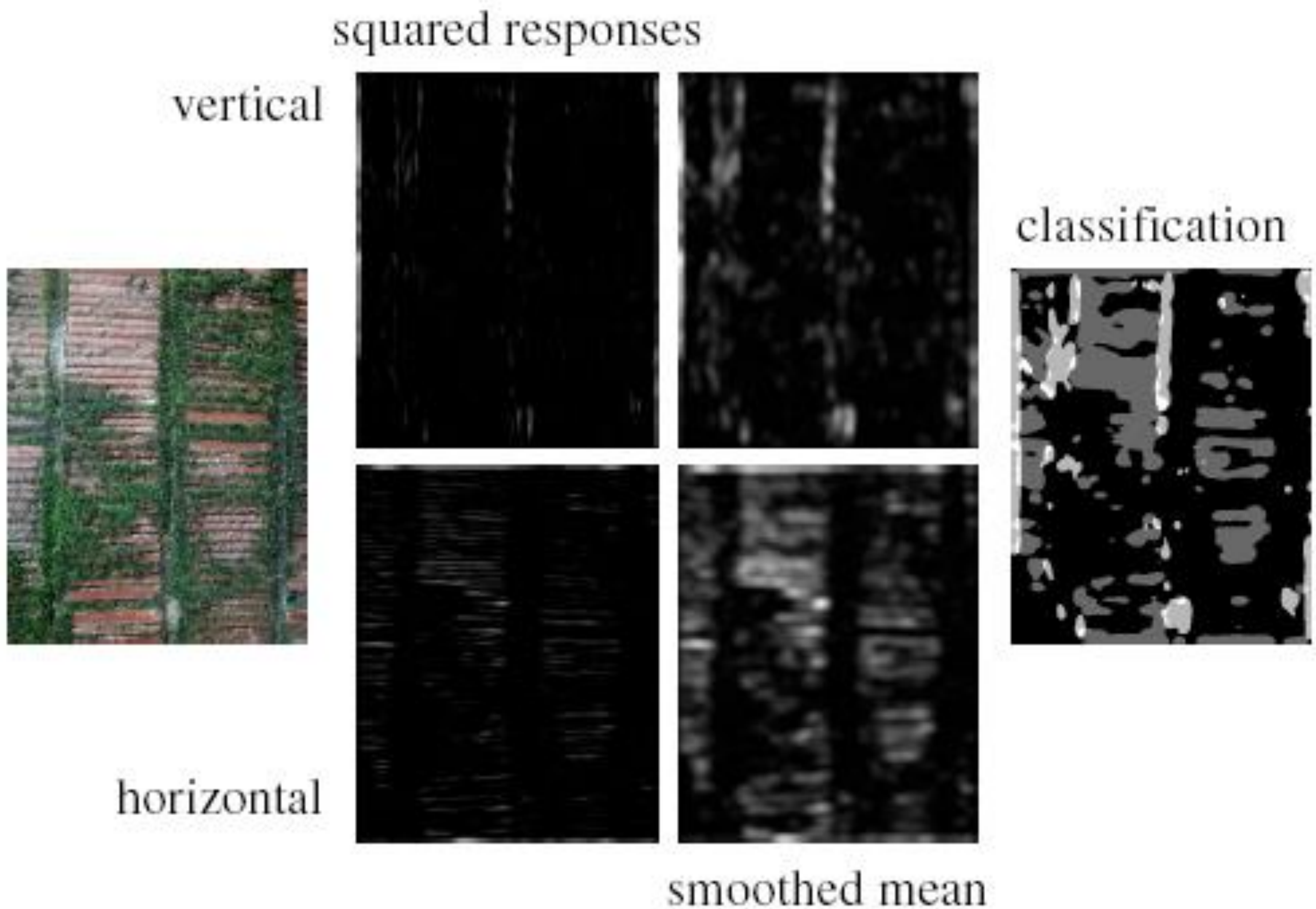


Figure 9.6 of D.A. Forsyth and J. Ponce, Computer Vision: A Modern Approach, Prentice Hall.

Different scales

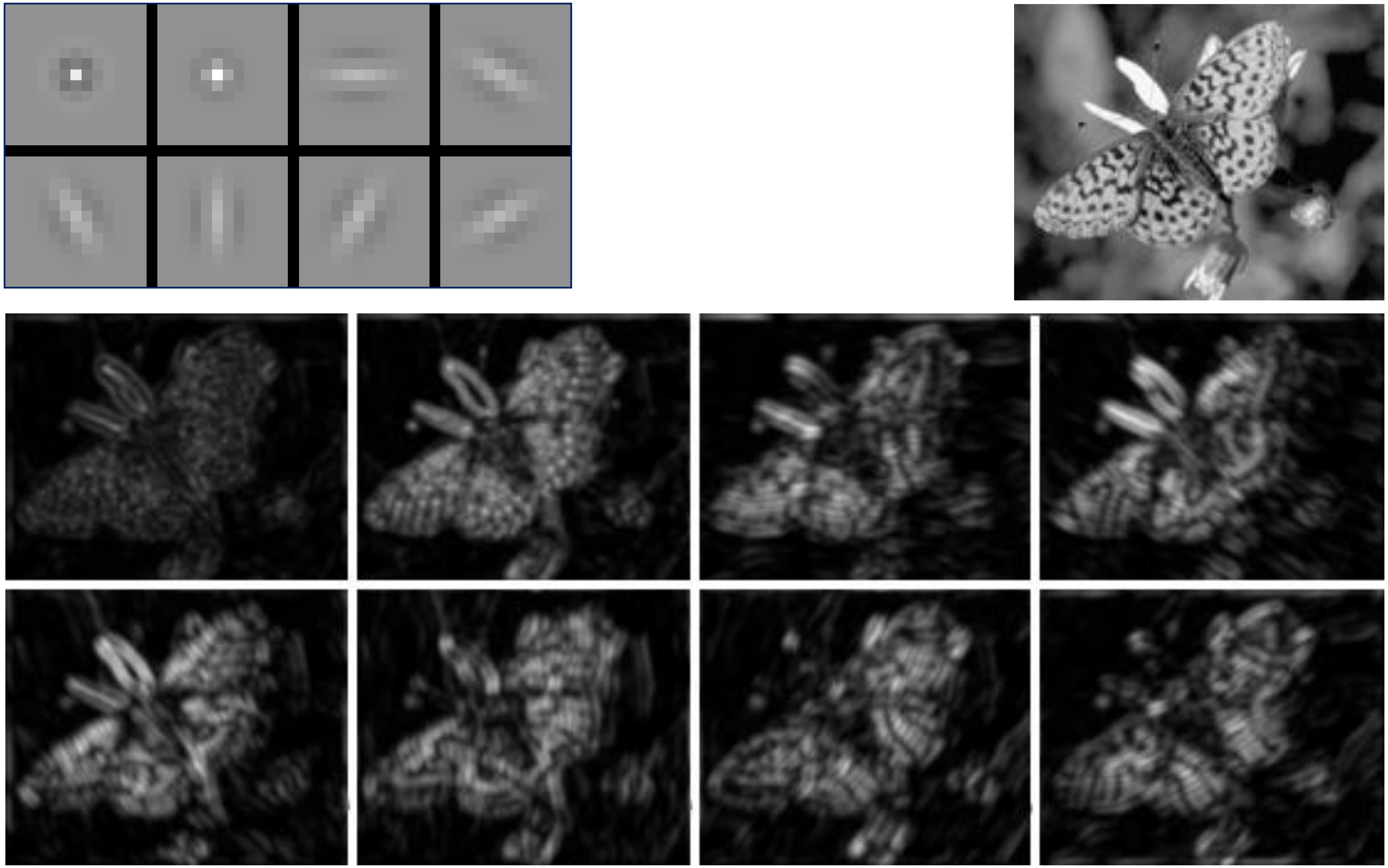
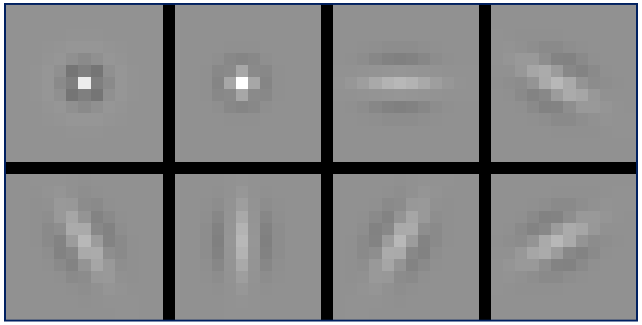
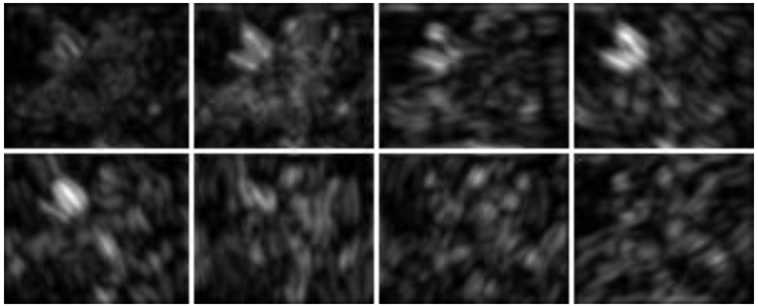


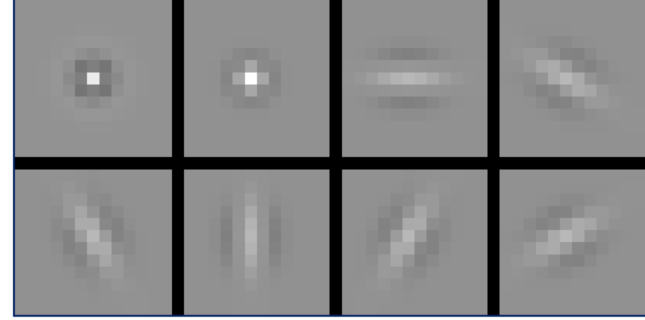
Figure 9.6 of D.A. Forsyth and J. Ponce, Computer Vision: A Modern Approach, Prentice Hall.

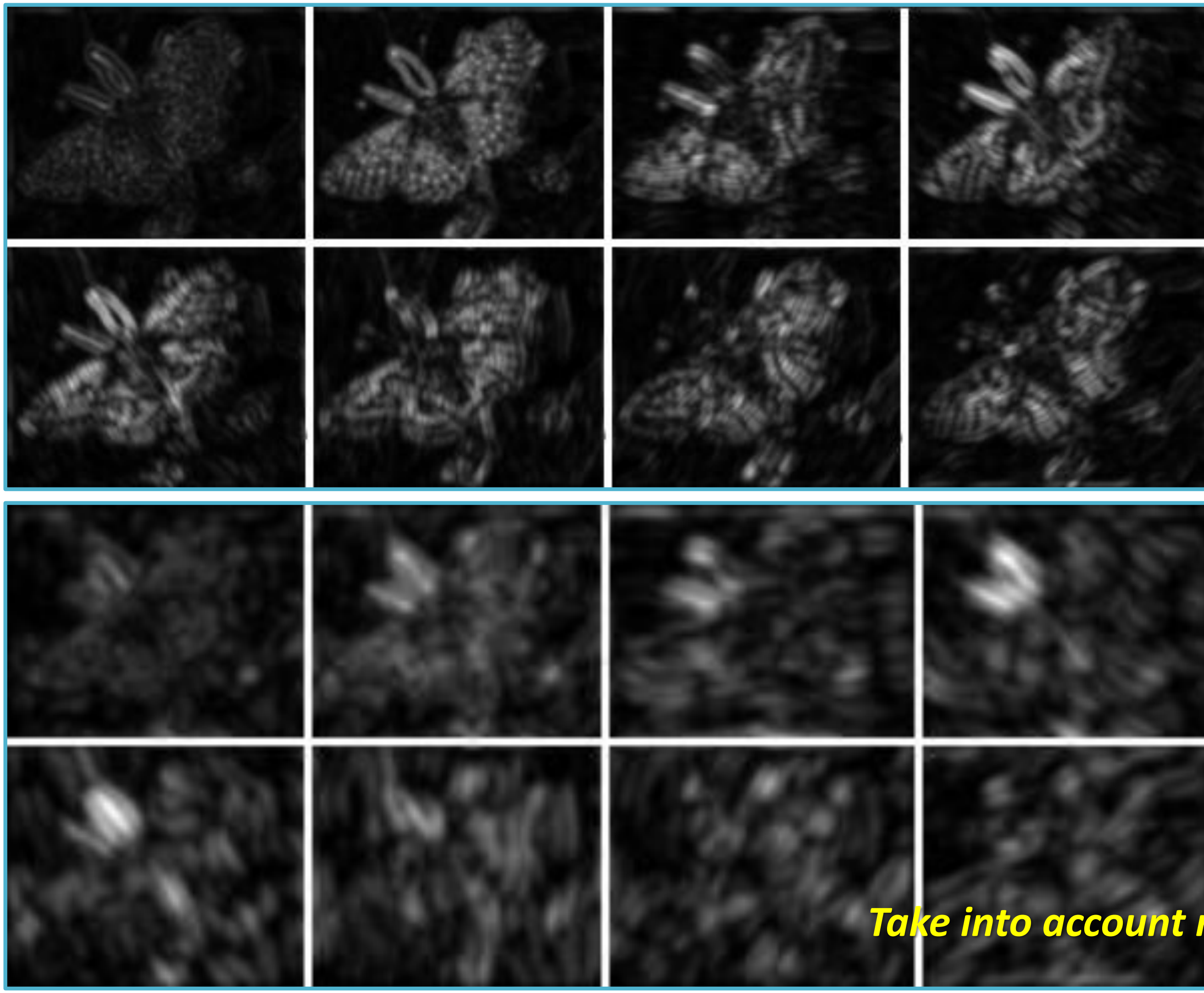
Different scales





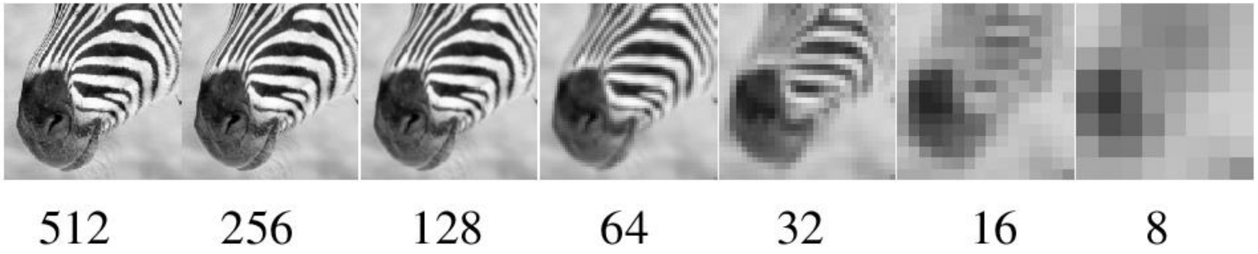
Responses of different scales





Take into account multiple scales !

Multiscale with a Gaussian pyramid



Laplacian operation

2D Laplacian operation

$$\triangle f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

	1	
1	-4	1
	1	

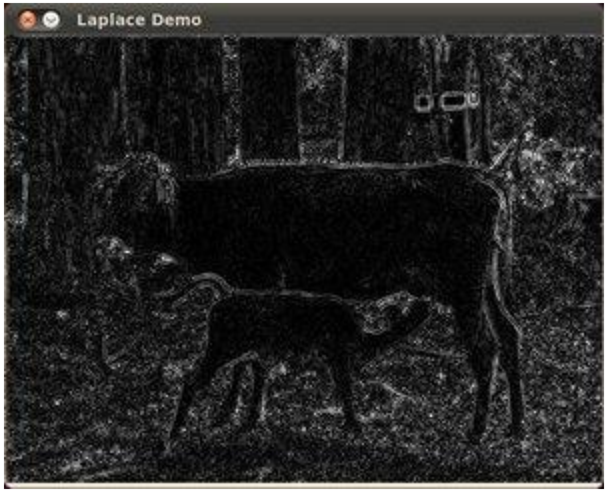


Fig. from https://docs.opencv.org/3.4/d5/db5/tutorial_laplace_operator.html

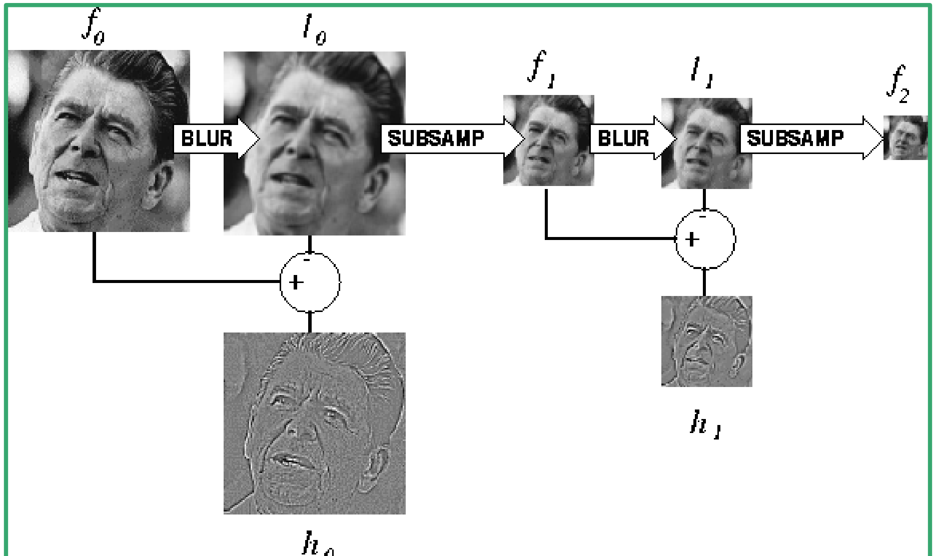
The Laplacian pyramid

- Building a Laplacian pyramid:
  - First, create a Gaussian pyramid
  - Take the difference between one Gaussian pyramid level and the next

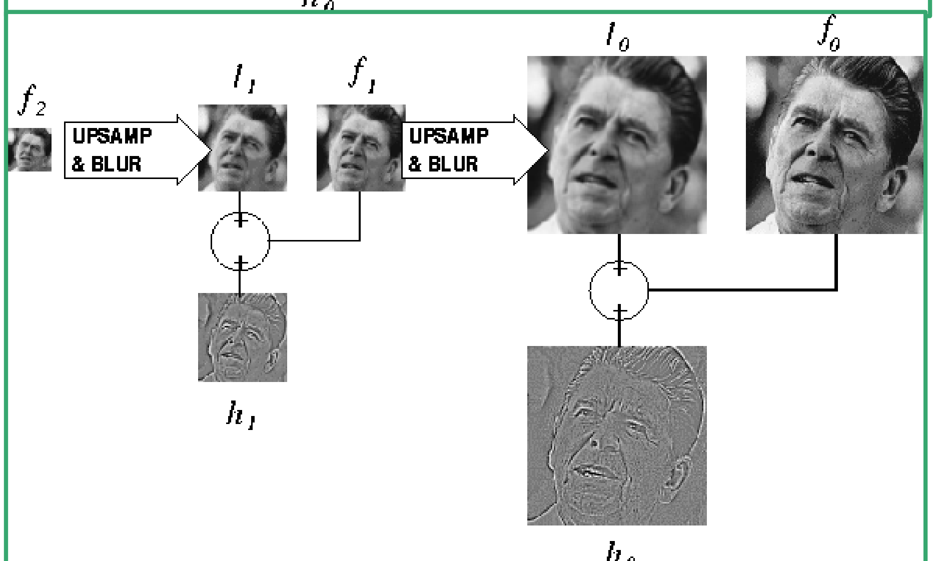
- A close approximation to the Laplacian.
- The coarsest level is the same as that in the Gaussian pyramid.
- Band pass filters: each level represents a different band of spatial frequencies

- Reconstructing the original image:
  - Reconstruct the Gaussian pyramid starting at top layer

The Laplacian pyramid



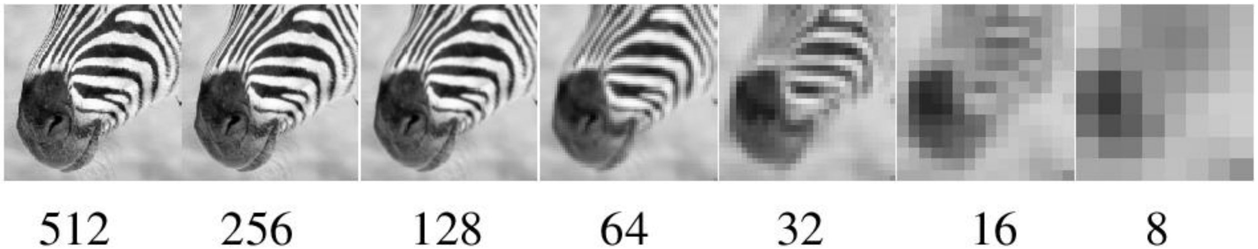
Create pyramid



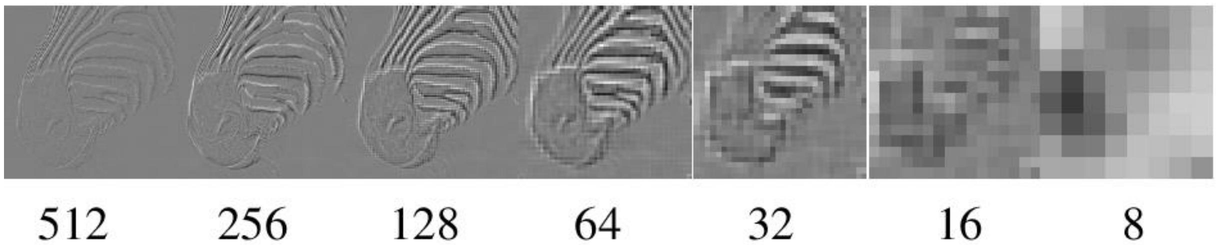
Collapse pyramid

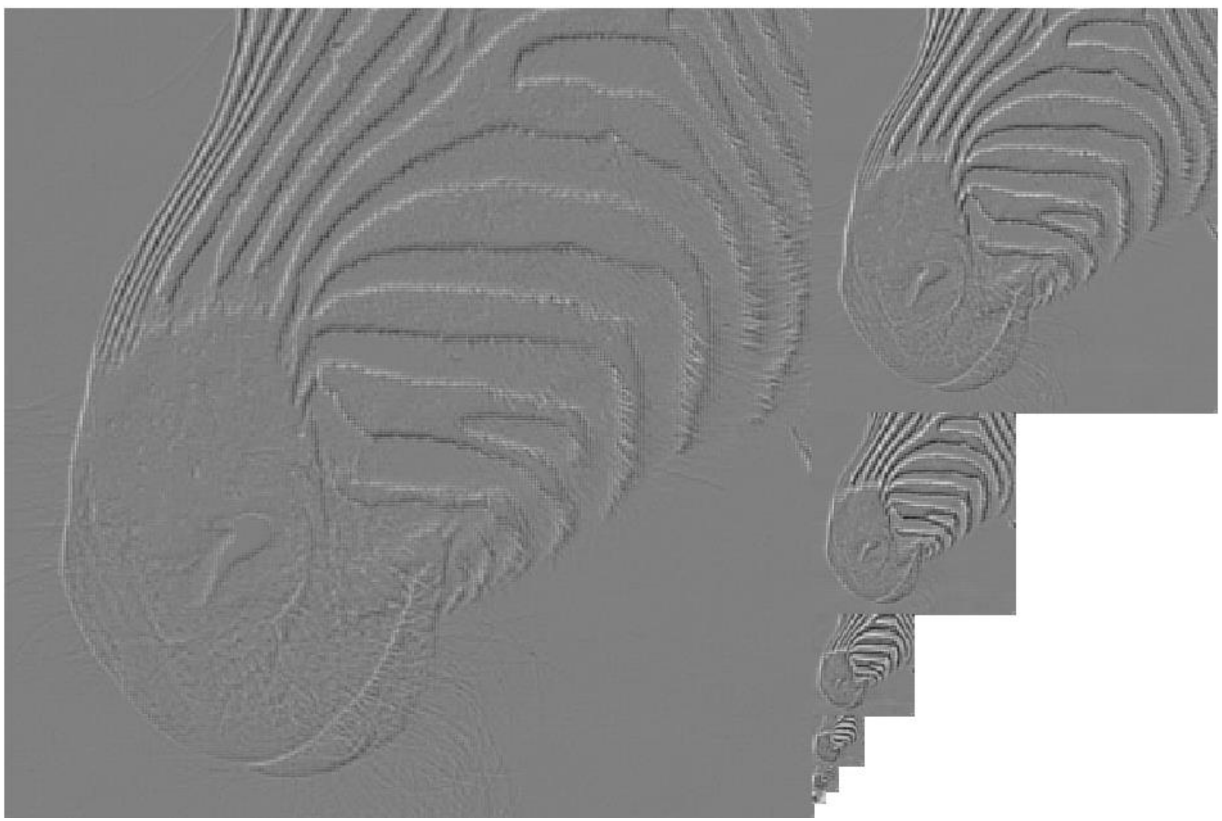
Figures from: http://sepwww.stanford.edu/~morgan/texturematch/paper_html/node3.html

Gaussian pyramid



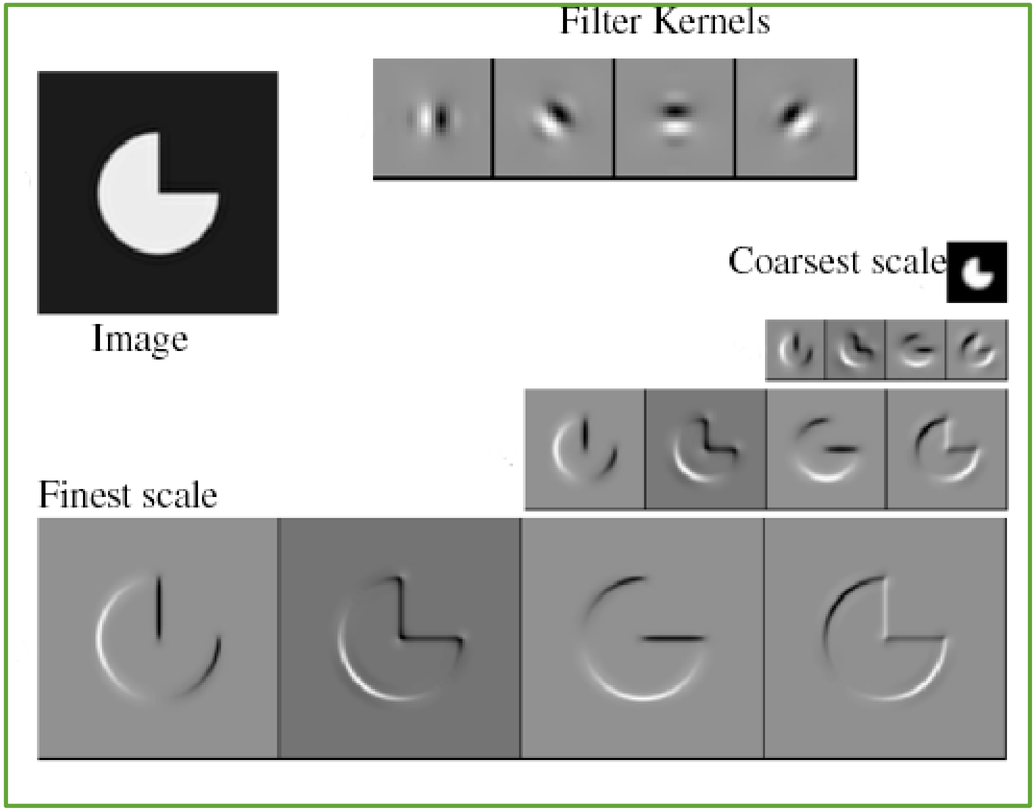
The corresponding Laplacian pyramid



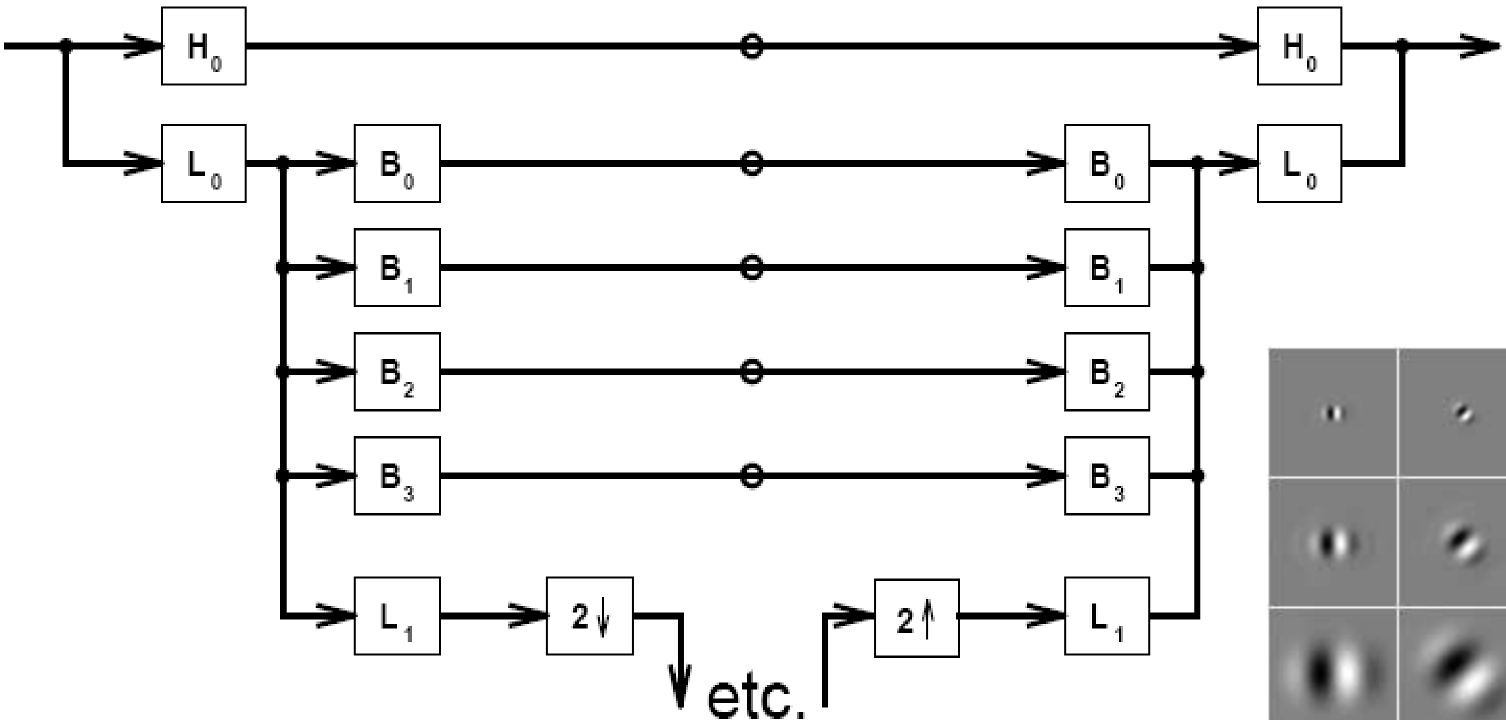


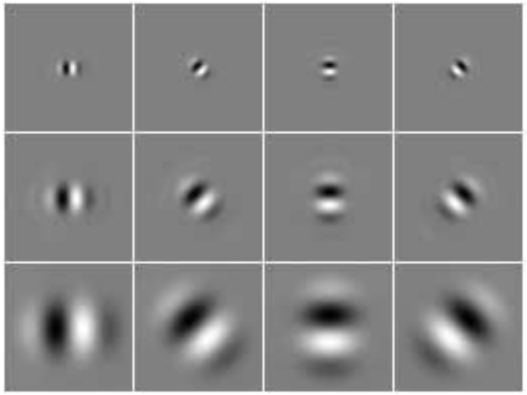
Oriented pyramids

- Laplacian pyramid is orientation independent.
  - Apply an oriented filter to determine orientations at each layer



Pyramid-based texture analysis/synthesis





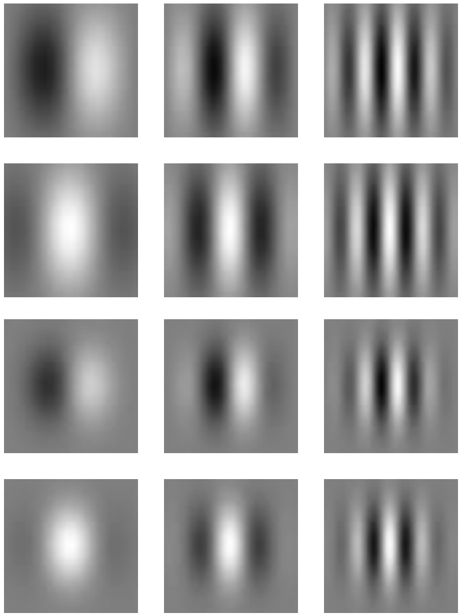
Gabor filters

- Gabor filters: Product of a Gaussian with sine or cosine
  - Considering local spatial frequency

- Top row shows anti-symmetric (or odd) filters, bottom row the symmetric (or even) filters.

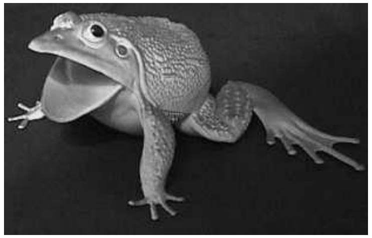
$$G_{antisymmetric}(x, y) = \sin(k_0 x + k_1 y) \exp - \left\{ \frac{x^2 + y^2}{2\sigma^2} \right\}$$

$$G_{Symmetric}(x, y) = \cos(k_x x + k_y y) \exp - \left\{ \frac{x^2 + y^2}{2\sigma^2} \right\}$$



Fundamental correspondence problems

- Feature matching for :
  - Scale
  - Rotation
  - Perspective
  - Occlusion
  - Illumination
  - Etc.



Scale Invariant Feature Transform

- **SIFT**: by D.G.Lowe, UBC

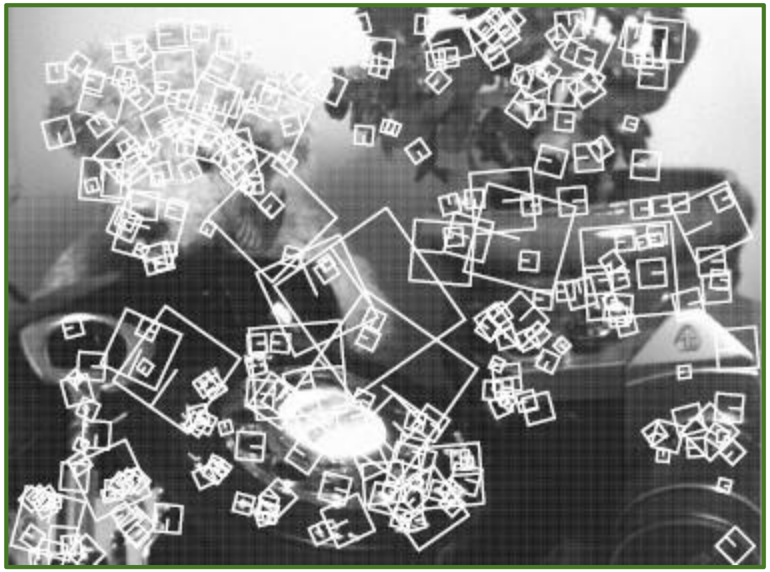
  - Transform image data into scale-invariant coordinates relative to local features

  - Conf.: “Object Recognition from Local Scale-invariant Features,” *Proc. Intl. Conf. Computer Vision (ICCV)*, vol.2, pp. 1150-1157, 1999. (citation 22162 at March 2022)

  - Journal: “Distinctive Image Features from Scale-invariant Keypoints,” *Intl. J. Computer Vision (IJCV)*, 60(2):91-110, 2004. (citation 66116 at March 2022)

SIFT (cont.)

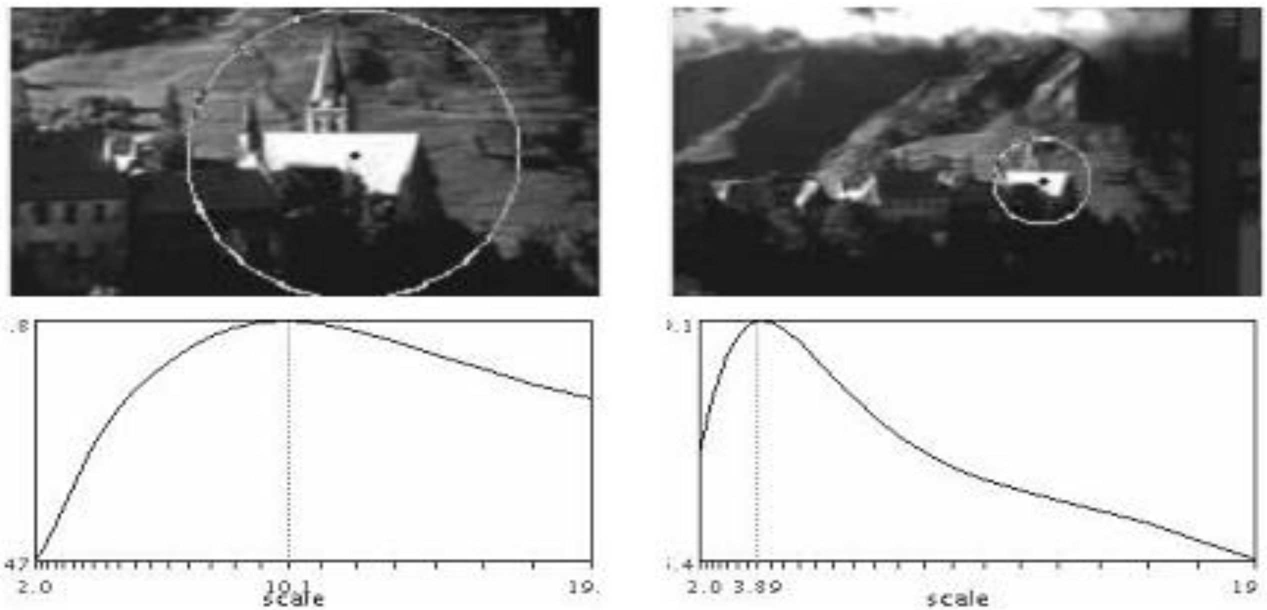
- Detection and description of local features.
- Procedures:
  - Detection of scale-space.
  - Keypoint localization.
  - Orientation assignment.
  - Local descriptor of keypoint.



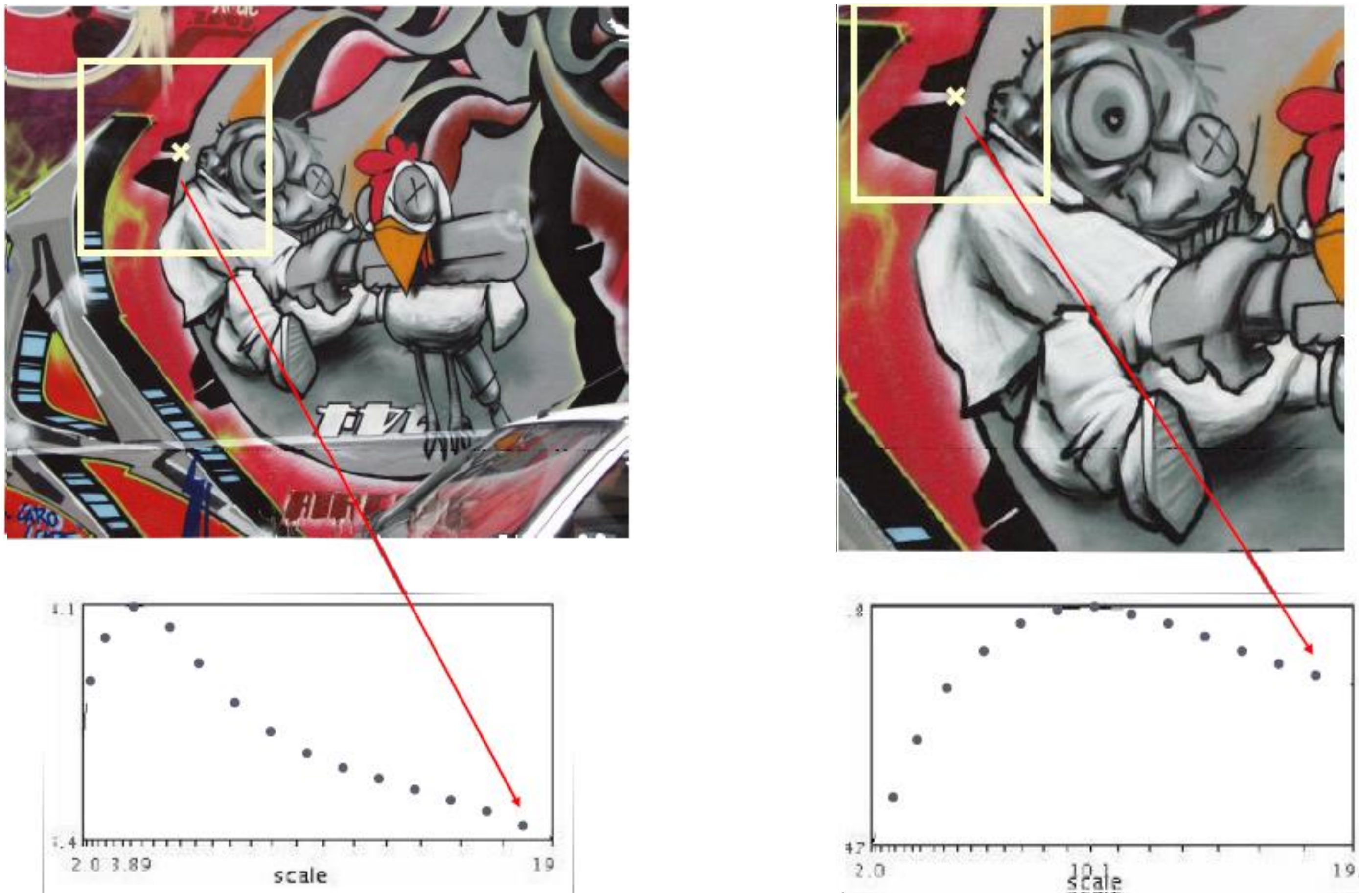
How to find the best scale-space?

- Mikolajczyk [2002] found that the maxima and minia of scaled-normalized Laplacian-of-Gaussian produce the best notation of scale.

$$\sigma^2 \nabla^2 G$$

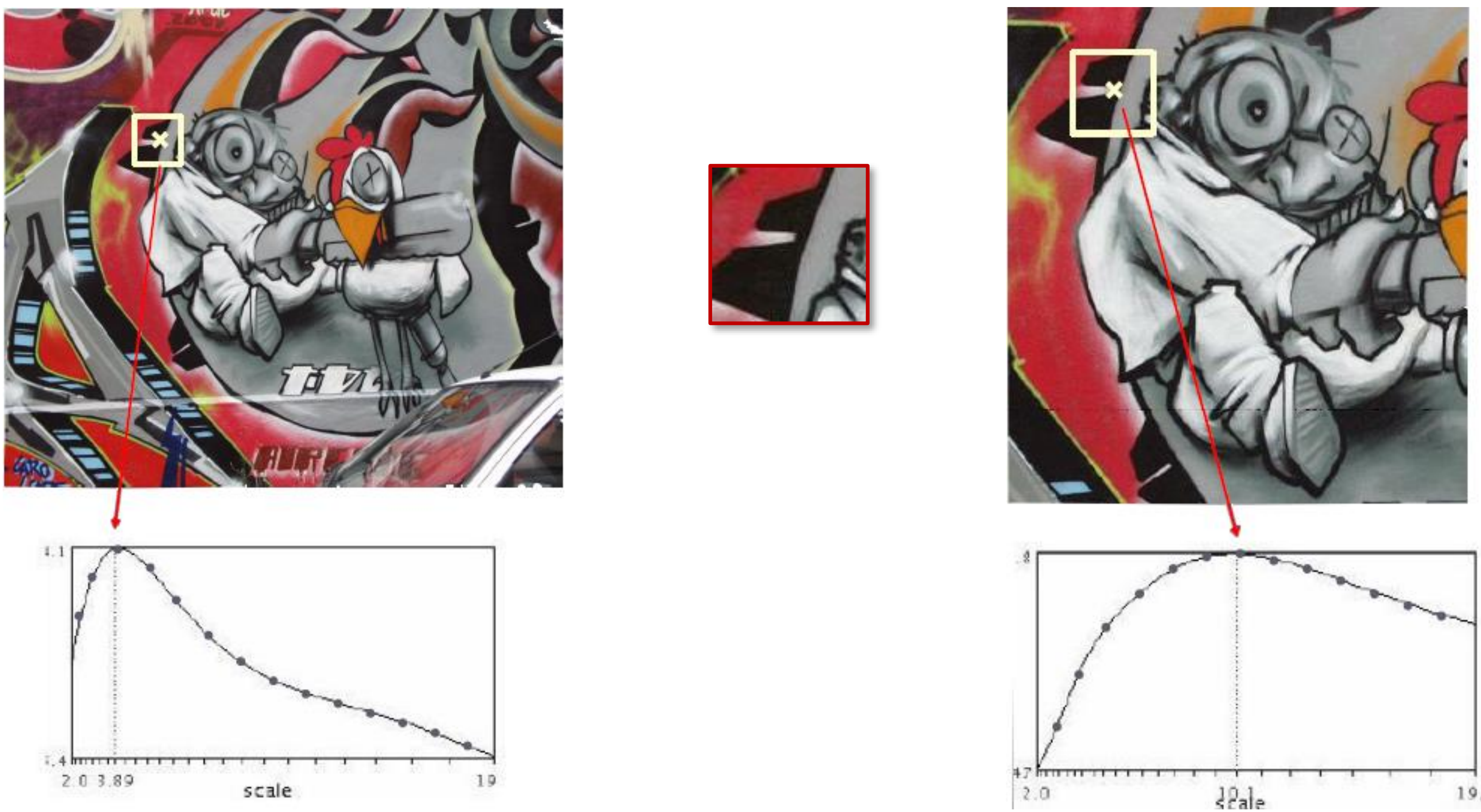


Automatic scale selection (Lindeberg et al., 1996)



Automatic scale selection

- Find the extreme of Laplacian or DoG.



Laplacian of Gaussian (LoG)

- Gaussian:

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

$$\frac{\partial G_\sigma(x,y)}{\partial x}=\frac{\partial}{\partial x}\left(\frac{1}{2\pi\sigma^2}e^{\frac{x^2+y^2}{-2\sigma^2}}\right)=\frac{1}{2\pi\sigma^2}\cdot\frac{\partial}{\partial x}e^{\frac{x^2+y^2}{-2\sigma^2}}=\frac{-x}{2\pi\sigma^4}\cdot e^{\frac{x^2+y^2}{-2\sigma^2}}$$

$$\frac{\partial^2 G_\sigma(x,y)}{\partial x^2}=\frac{\partial}{\partial x}\left(\frac{-x}{2\pi\sigma^4}\right)\cdot e^{\frac{x^2+y^2}{-2\sigma^2}}+\frac{-x}{2\pi\sigma^4}\cdot\frac{\partial}{\partial x}\left(e^{\frac{x^2+y^2}{-2\sigma^2}}\right)$$

$$=\frac{-1}{2\pi\sigma^4}\cdot e^{\frac{x^2+y^2}{-2\sigma^2}}+\frac{-x}{2\pi\sigma^4}\cdot e^{\frac{x^2+y^2}{-2\sigma^2}}\cdot\frac{x}{-\sigma^2}=\left(\frac{x^2-\sigma^2}{2\pi\sigma^6}\right)\cdot e^{\frac{x^2+y^2}{-2\sigma^2}}$$

$$\frac{\partial^2 G_\sigma(x,y)}{\partial y^2}=\left(\frac{y^2-\sigma^2}{2\pi\sigma^6}\right)\cdot e^{\frac{x^2+y^2}{-2\sigma^2}}$$

- LoG:

$$\mathrm{LoG}\equiv\Delta G_\sigma(x,y)=\frac{\partial^2 G_\sigma(x,y)}{\partial x^2}+\frac{\partial^2 G_\sigma(x,y)}{\partial y^2}=\left(\frac{x^2+y^2-2\sigma^2}{2\pi\sigma^6}\right)\cdot e^{\frac{x^2+y^2}{-2\sigma^2}}$$

Using Lapacian or DoG

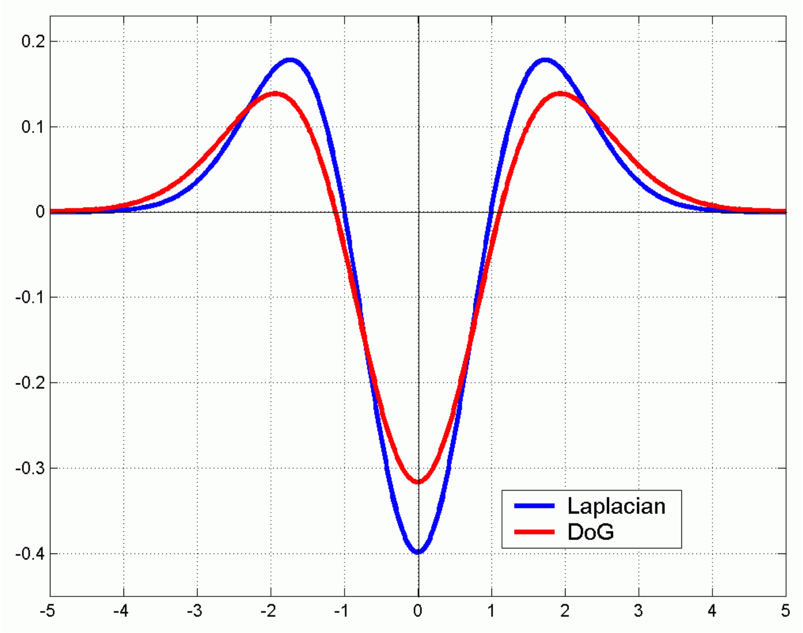
$$L = \sigma^2 \left(G_{xx}(x, y, \sigma) + G_{yy}(x, y, \sigma) \right)$$

Goal (Laplacian)

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

(Difference of Gaussians)

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



Approximate of LoG

- Instead of direct evaluation, LoG can be approximated as:

$$\sigma\nabla^2 G = \frac{\partial G}{\partial \sigma} \approx \frac{G(k\sigma) - G(\sigma)}{k\sigma - \sigma}$$

$$\begin{aligned}
\frac{\partial G}{\partial \sigma} &= \frac{-1}{\pi\sigma^3} e^{-\frac{x^2+y^2}{2\sigma^2}} + \frac{x^2+y^2}{2\pi\sigma^5} e^{-\frac{x^2+y^2}{2\sigma^2}} \\
&= \frac{-2\sigma^2 + x^2 + y^2}{2\pi\sigma^5} e^{-\frac{x^2+y^2}{2\sigma^2}} \\
&= \sigma \frac{-2\sigma^2 + x^2 + y^2}{2\pi\sigma^6} e^{-\frac{x^2+y^2}{2\sigma^2}} \\
&= \sigma\nabla^2 G
\end{aligned}$$

- We only calculate the difference-of-Gaussian if *k* is a constant, e.g. √2.

$$G(k\sigma) - G(\sigma) \approx (k-1)\sigma^2\nabla^2 G$$

Scale-space construction

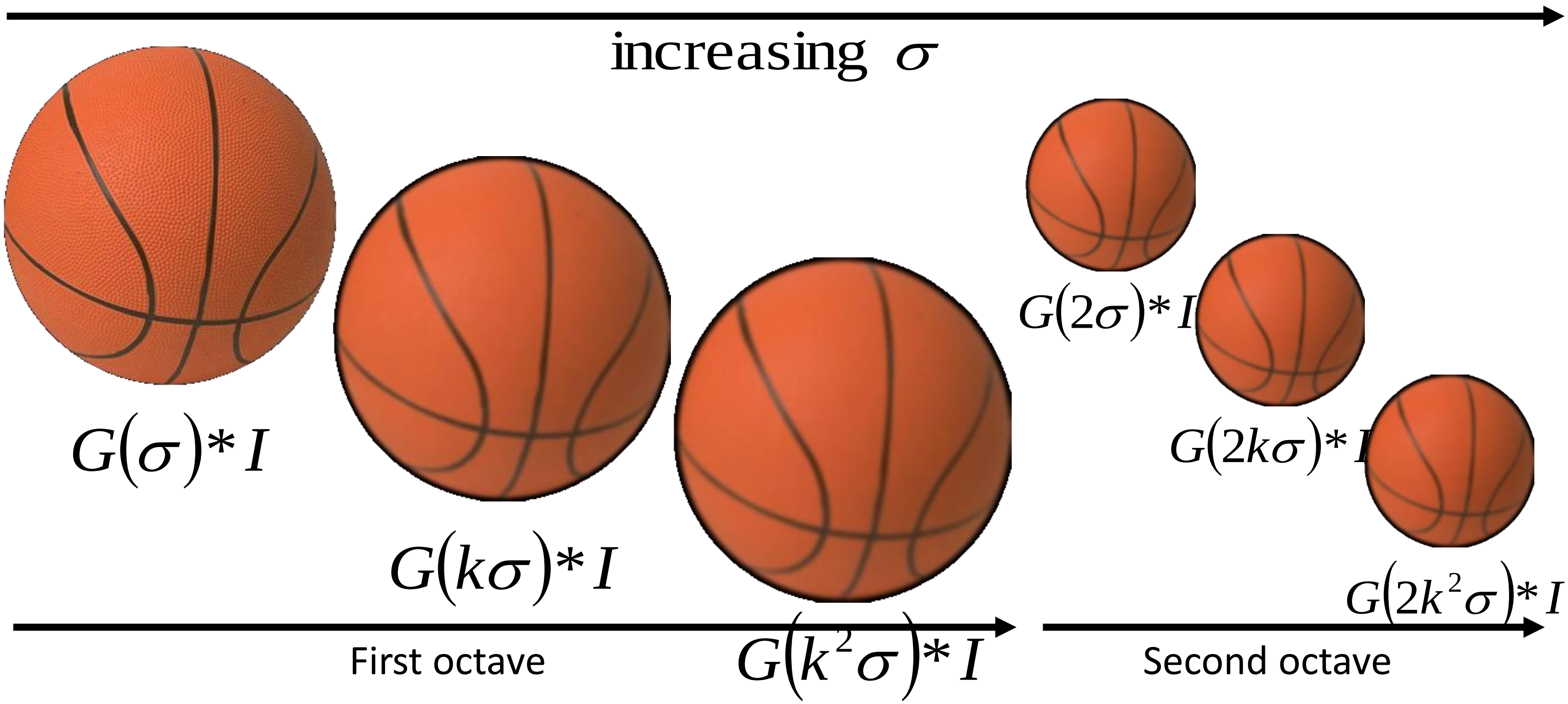
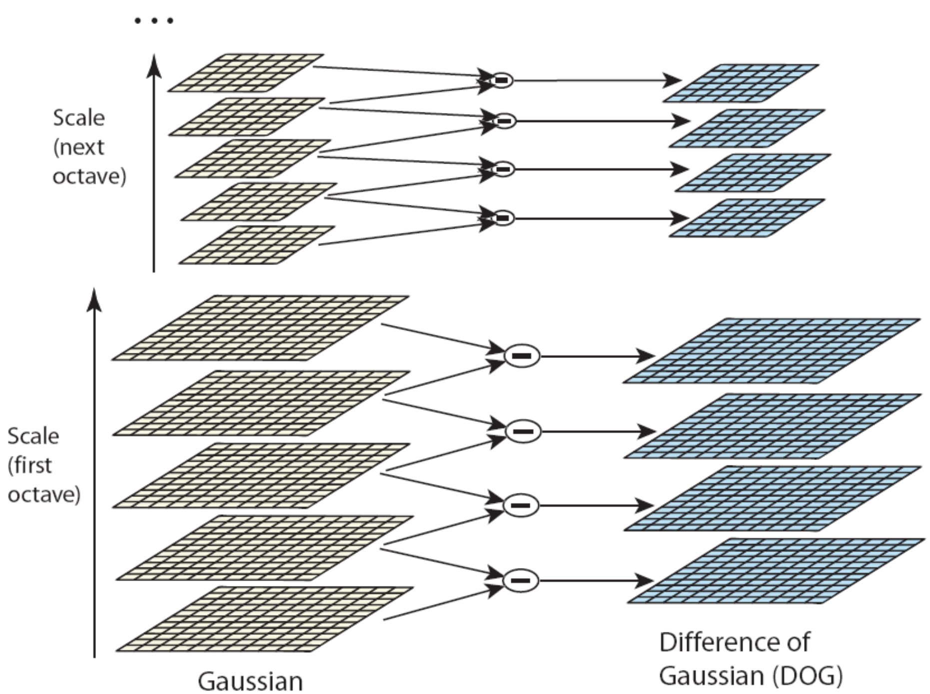
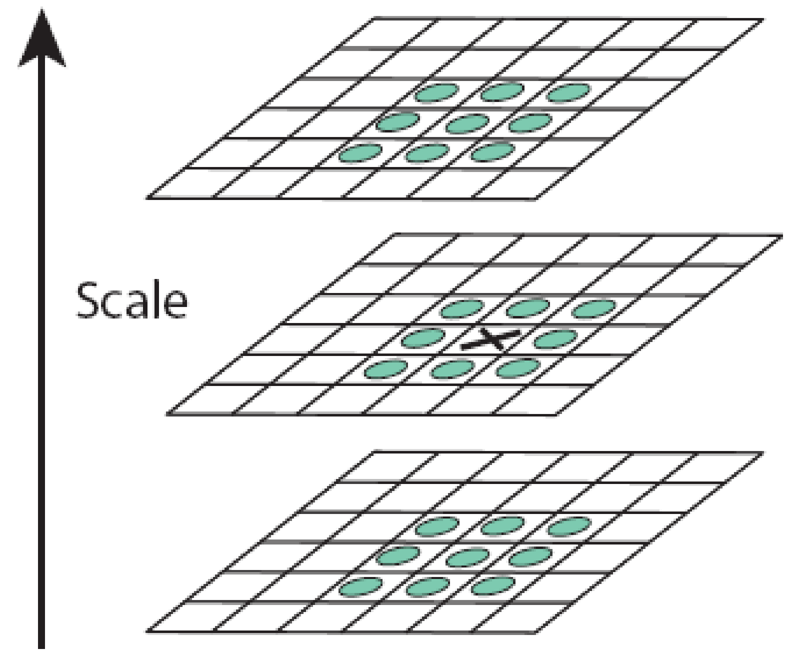


Figure from O. Pele, the presentation slides of “SIFT: Scale Invariant Feature Transform.”

Finding the extremes

- Construct difference of Gaussian(DOG) first.
- Choose all extremes within 3x3x3 neighborhood.

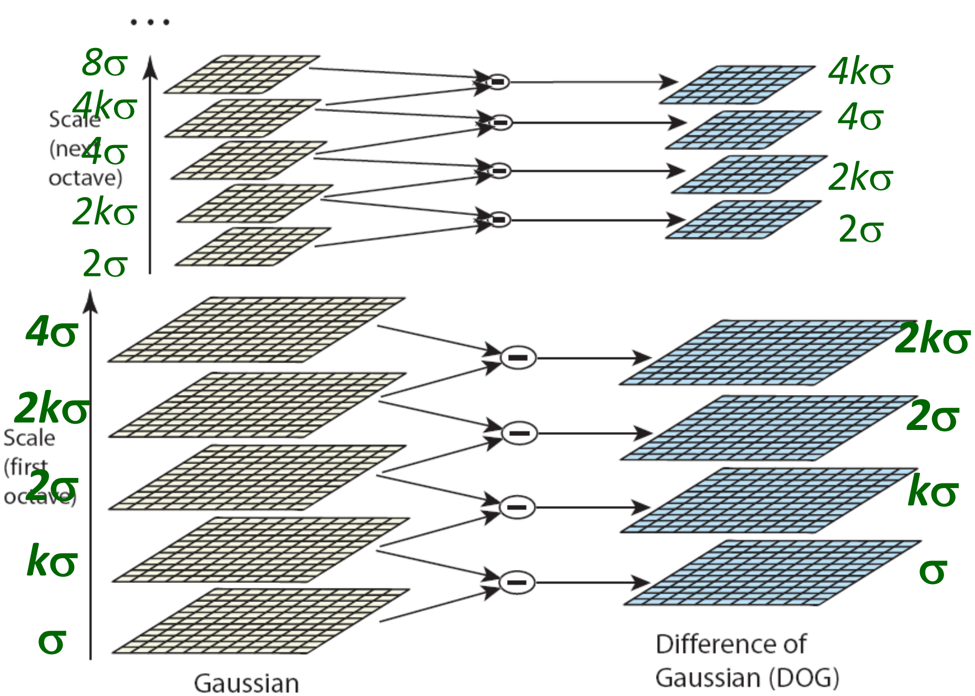


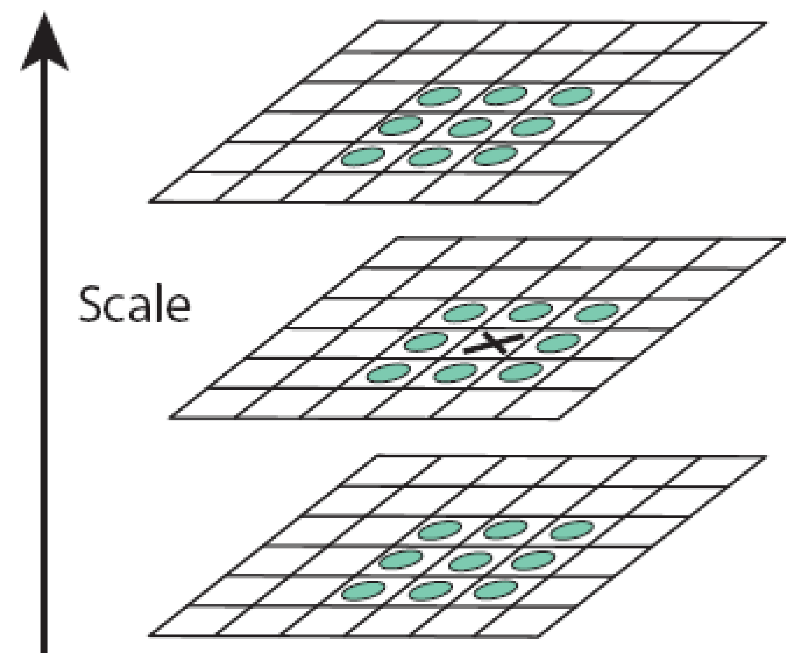


$$
\begin{aligned}
D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\
&= L(x, y, k\sigma) - L(x, y, \sigma).
\end{aligned}
$$

Finding the extremes

- s+3 filtered images are evaluated in an octave.
- For instance, $s = k^2 = 2$,



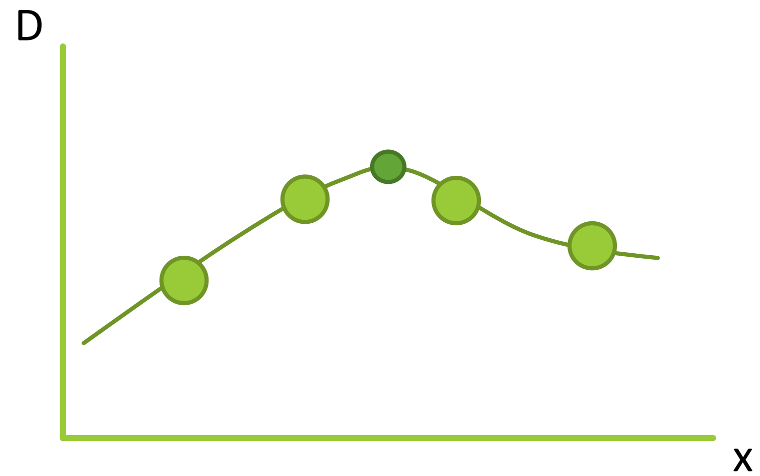


Accurate keypoint localization

- There are still plenty of points, some of them are not good enough.
- The locations of keypoints may be not accurate.
  - Pixel-level accuracy.
- Eliminating edge or improper points.

Keypoint localization

- While approximating scale-space function, D(x, y, σ) by quadratic Talyor expansion,



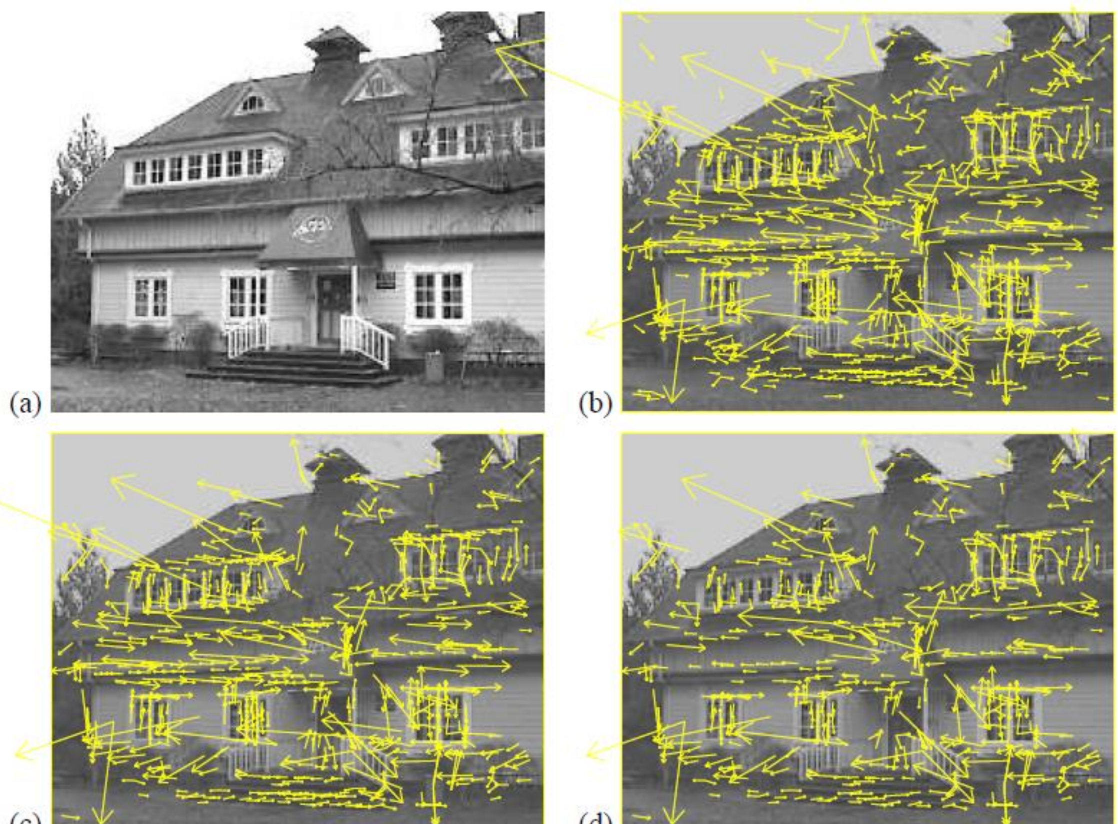
$$D(\vec{x}) = D + \frac{\partial D^T}{\partial \vec{x}} \vec{x} + \frac{1}{2} \vec{x}^T \frac{\partial^2 D^T}{\partial \vec{x}^2} \vec{x}$$

- Determine the location of extremum by $\frac{\partial D(x)}{\partial x} = 0$

$$\hat{x} = -\frac{\partial^2 D}{\partial \vec{x}^2}^{-1} \frac{\partial D}{\partial \vec{x}} \qquad D(\hat{x}) = D + \frac{1}{2} \frac{\partial D^T}{\partial \vec{x}} \hat{x}$$

Keypoint: Removing unstable extremes

- Remove $D(\hat{x})$ smaller than 0.03 (image values in [0,1]) .



Original image

832 local extremes

729 keypoints (stable D)

536 keypoints (valid ratio of eigen values)

Keypoints: Eliminating edge points

- Reject points with strong edge response in one direction only.
- Similar to Harris corner – remove points with a large principal curvature across the edge but a small one in the perpendicular direction.
- The principal curvatures can be calculated from a Hessian function

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

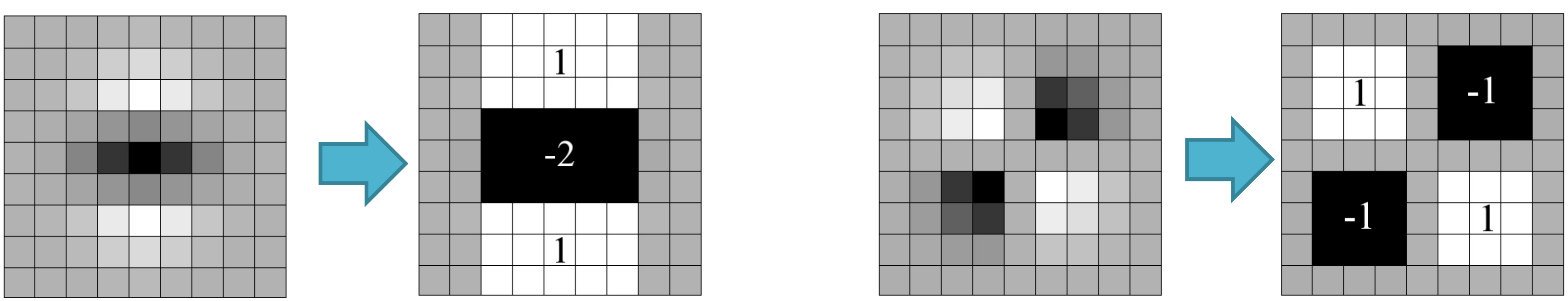
Keypoint localization

- Harris corner use the 2nd order moment matrix.

$$A = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$$

- SIFT and SURF uses the Hessian matrix (principal curvatures) for efficiency.

$$H(x,\sigma) = \begin{bmatrix} L_{xx}(x,\sigma) & L_{xy}(x,\sigma) \\ L_{xy}(x,\sigma) & L_{yy}(x,\sigma) \end{bmatrix} \approx \begin{bmatrix} D_{xx}(x,\sigma) & D_{xy}(x,\sigma) \\ D_{xy}(x,\sigma) & D_{yy}(x,\sigma) \end{bmatrix}$$



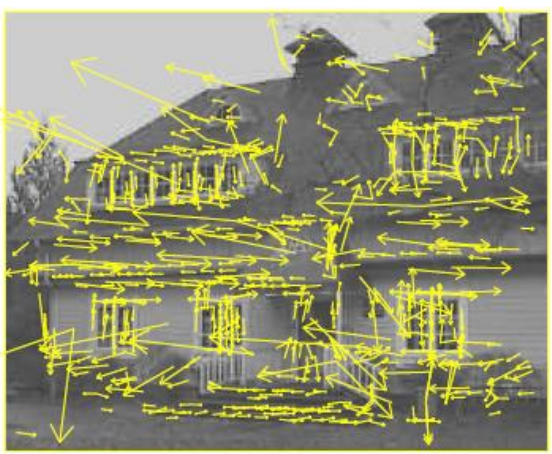
L_{yy} D_{yy} L_{xy} D_{xy}

Keypoints: Eliminating edge points (cont.)

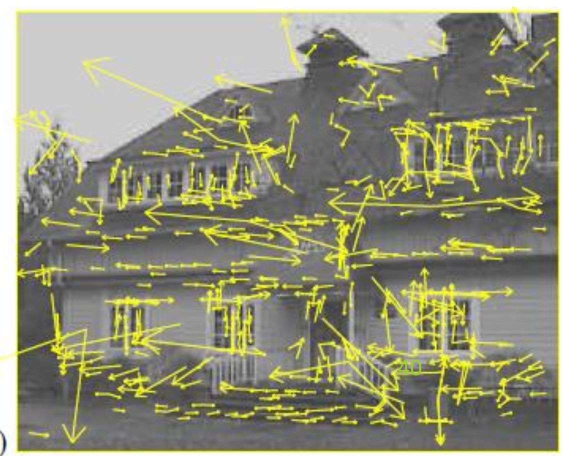
- α,β are the large and small eigen values of H.
- Tr(H) = α+β, Det(H) = αβ ，r = α/β
- To check if ratio of principal curvatures is below some threshold, e.g. r_{th}=10, check:

$$\frac{Tr(H)^2}{Det(H)} = \frac{(r+1)^2}{r} < \frac{(r_{th}+1)^2}{r_{th}}$$

729 keypoints
(stable D)



(c)



(d)

536 keypoints
(valid ratio of
eigen values)

Local descriptor

- After extracting keypoints, the next goal is to find an appropriate descriptor for the local area.

- Should be robust to:
  - Rotation/Perspective transformation
  - Illumination change
  - Noise

- Should be compact and easily for matching.

Orientation assignment

- The keypoint descriptor can be represented relative to this orientation and therefore achieve invariance to image rotation.

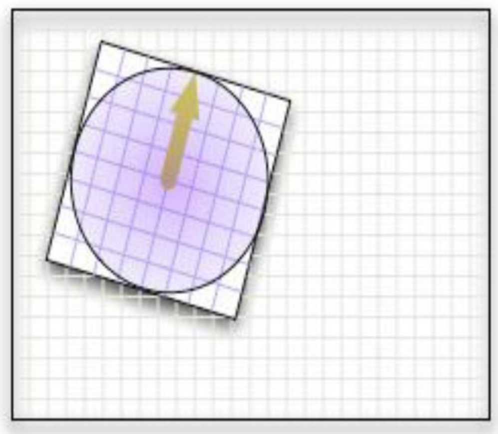


Figure from: Jonas Hurrelmann, Ofir Pele's slides.

- Compute magnitude and orientation on the Gaussian smoothed images *L*:

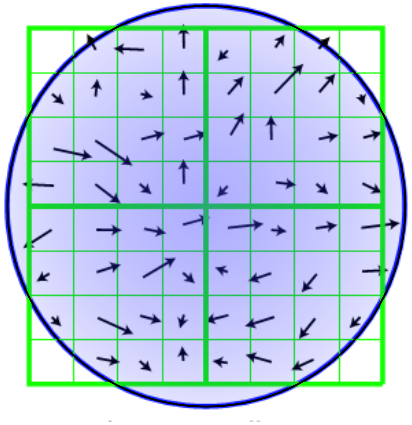
$$m(x,y)=\sqrt{\left(L(x+1,y)-L(x-1,y)\right)^2+\left(L(x,y+1)-L(x,y-1)\right)^2}$$

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

Orientation assignment (cont.)

- Consider a window a σ, 1.5 times of the keypoint scale.

- An orientation histogram with 36 bins for 360º.
  - Weighted by magnitude and Gaussian window.
  - Any peak within 80% of the highest peak is used to create a keypoint with that orientation.
  - Near 15% assigned multiple orientations, but contribute significantly to the stability.



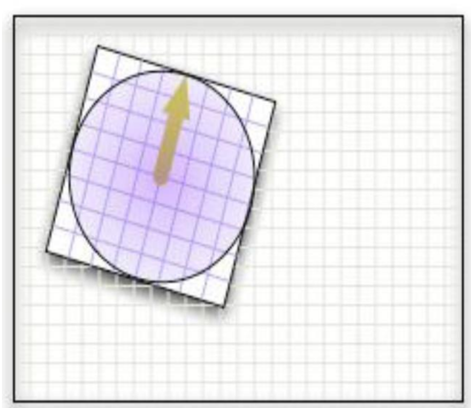
- More accurate orientation by parabola fitting of 3 histogram values closest to each peak.

SIFT descriptor

- Each point has position**(*x, y*)**, scale ***σ***, gradient magnitude ***m***, orientation ***ϑ***.
- Local feature descriptor:
  - Based on 16*16 patches
  - 4*4 subregions
  - 8 bins in each subreg

The magnitudes are also weighted by a Gaussian function.

4*4*8=128 dimensions in total



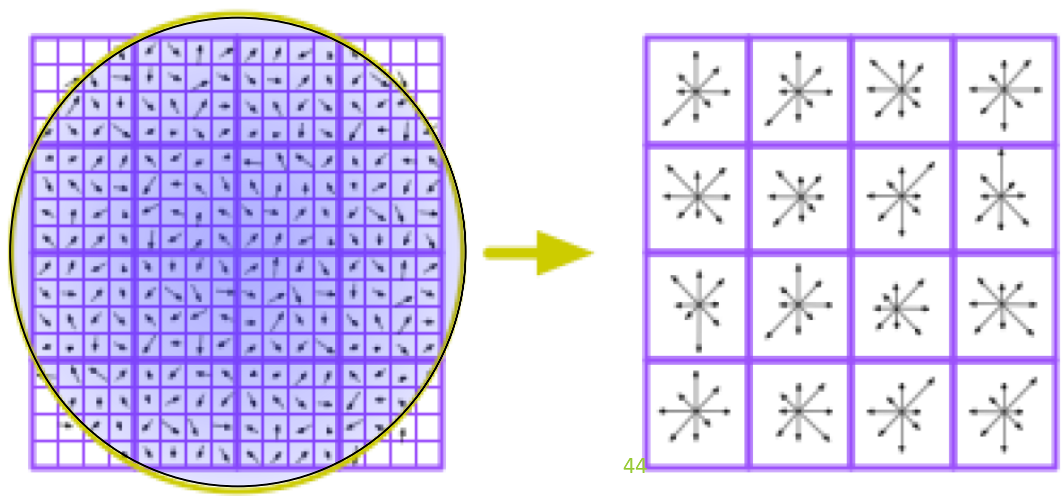


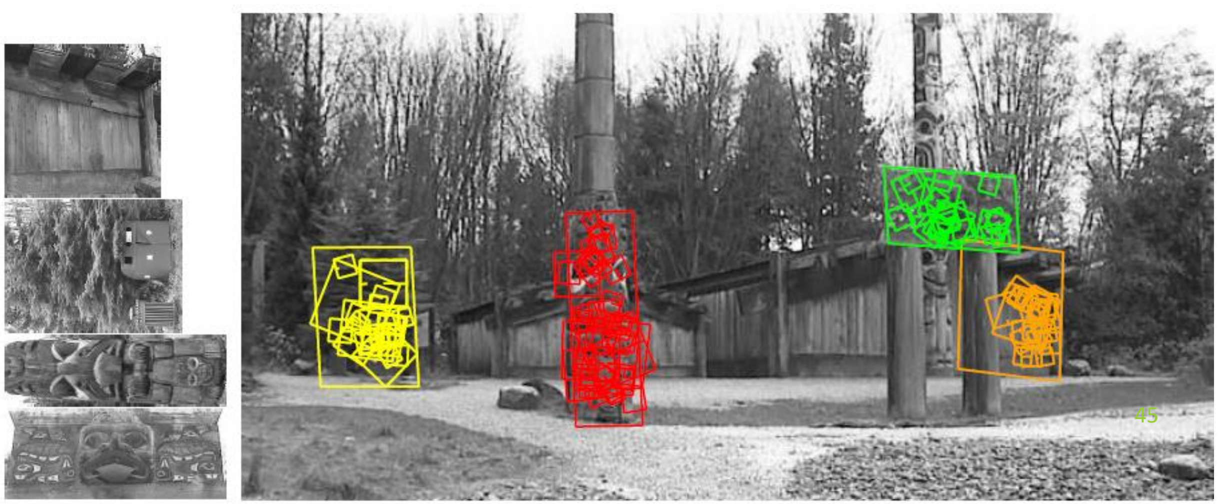
Image gradients

Keypoint descriptor

Figure from: Jonas Hurrelmann, Ofir Pele's slides.

Application: Object recognition

- The SIFT features of training images are extracted and stored.
- For a query image
  - Extract SIFT feature
  - Efficient nearest neighbor indexing
  - 3 keypoints, Geometry verification (affine)



Application: Object recognition

Applications: Image alignment

[Brown & Lowe 2003]