— CM50248 — 2017/2018 —

Visual Understanding 1

Dr Christian Richardt

Fourier transform (recap)

- The **Fourier transform** of a function $f : \mathbb{R} \to \mathbb{R}$ is given by:

$$F(\omega) = \mathcal{F}[f](\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x) \mathrm{e}^{-i\omega x} dx$$

- The **inverse Fourier transform** restores f from F:

$$f(x) = \mathcal{F}^{-1}[F](\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} F(\omega) \mathrm{e}^{i\omega x} d\omega$$

- Note the similarity between the Fourier transform and its inverse.
- We use $F(\omega)$ and $\mathcal{F}[f](\omega)$interchangeably to refer to the representation of $f(x)$in the frequency (or Fourier) domain.

Convolution theorem (recap)

- The convolution $f * g$ of two functions f and g is given by:

$$(f * g)(x) = \int_{-\infty}^{\infty} f(x - y)g(y)\, dy$$

- **Convolution theorem:** $\mathcal{F}[f * g](\omega) = \sqrt{2\pi}\mathcal{F}[f](\omega) \cdot \mathcal{F}[g](\omega)$
  - Convolution becomes multiplication in the Fourier domain
  - Afterwards, transform results back to spatial domain
- As we will see, this can save a lot of time for convolutions with large kernels
- Reciprocal convolution theorem: $\mathcal{F}[f \cdot g](\omega) = \sqrt{2\pi}\mathcal{F}[f](\omega) * \mathcal{F}[g](\omega)$

Fast Fourier transform (FFT, recap)

- Key insight: divide and conquer
  - Split an input of size *N* into two inputs of size N/2 each
  - Repeat until inputs of size 1 is reached
  - This is has $\mathcal{O}(\log N)$complexity: need this many levels of nesting
- Advantages:
  - Very efficient: $\mathcal{O}(N \log_2 N)$
  - Implemented in many libraries (e.g. FFTW used by MATLAB)
- Disadvantages:
  - Requires inputs of size $N = 2^k$, so input may need padding

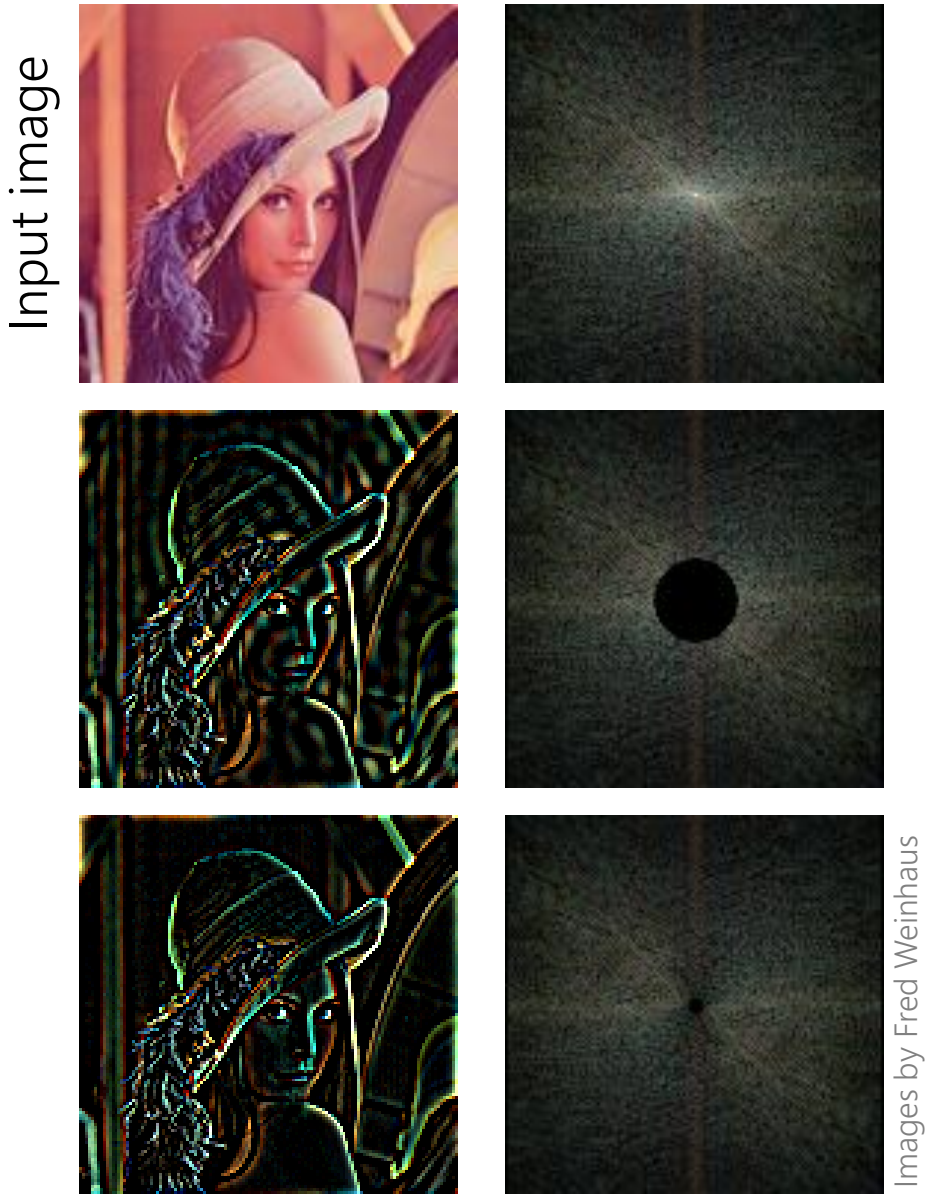
Low-pass filter (recap)

- passes signals with frequencies lower than a threshold
  - Removes fine details and hence blurs an image
- Example using hard cut-off in frequency space:
  - Ringing artefacts in image space
- Better low-pass filter: Gaussian blur
  - Removes ringing artefacts

Images by Fred Weinhaus

High-pass filter (recap)

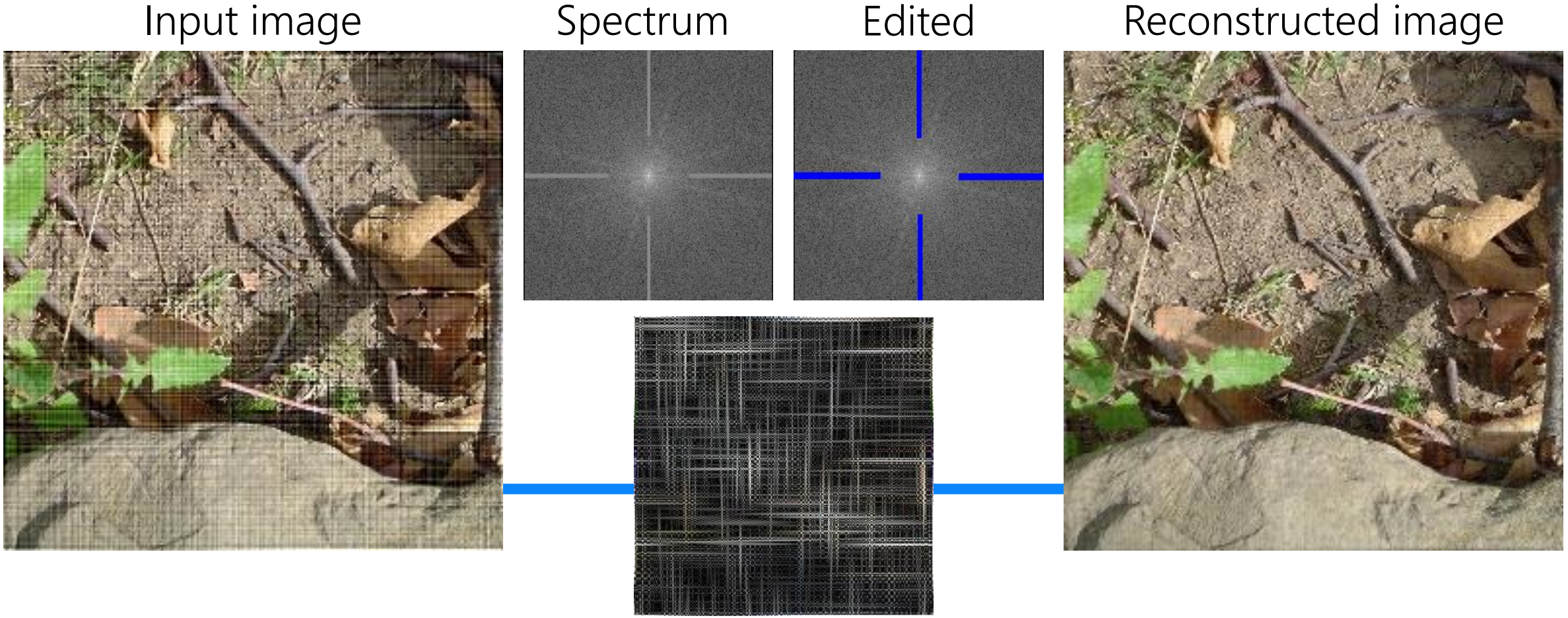
- passes signals with frequencies higher than a threshold
  - Removes low frequencies (~overall shape)
  - get edge image
- Example using hard cut-off in frequency space:
  - Ringing artefacts in image space
- Better high-pass filter: 1–Gaussian blur
  - Removes ringing artefacts



Images by Fred Weinhaus

Structured noise removal (recap)

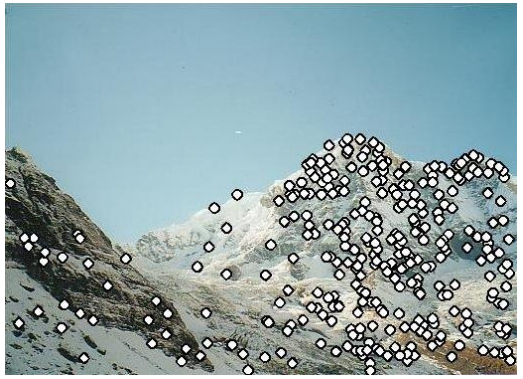
- Discard regions in spectrum associated with patterned noise



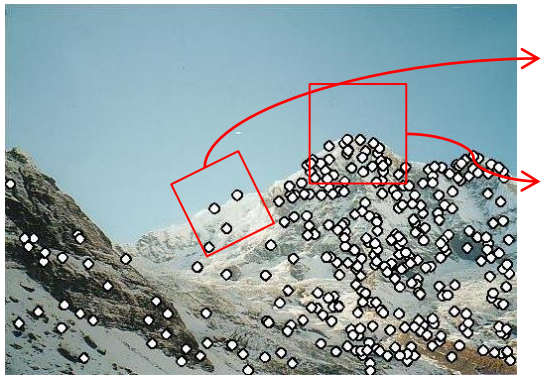
Images by Fred Weinhaus

Local invariant features

1. **Detection:**
   Identify the interest points



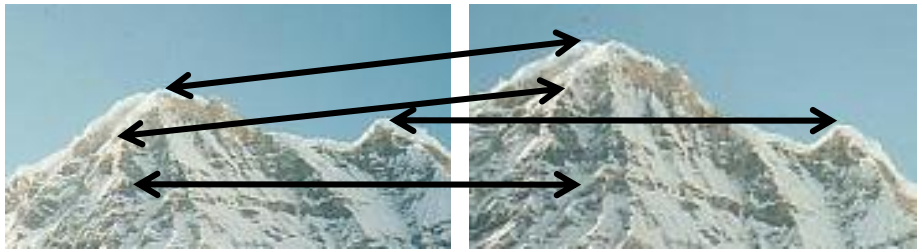
2. **Description:**
   Extract vector feature descriptor surrounding each interest point.



$$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$$

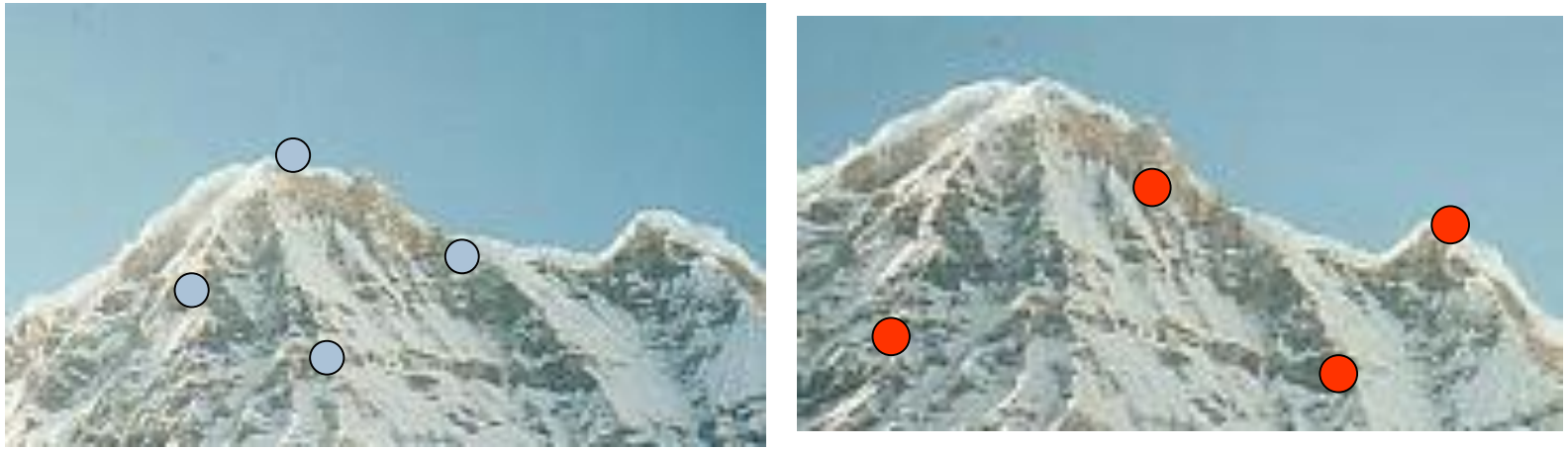
$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$

3. **Matching:**
   Determine correspondence between descriptors in 2 views



Slide by Kristen Grauman

Goal: repeatable detector

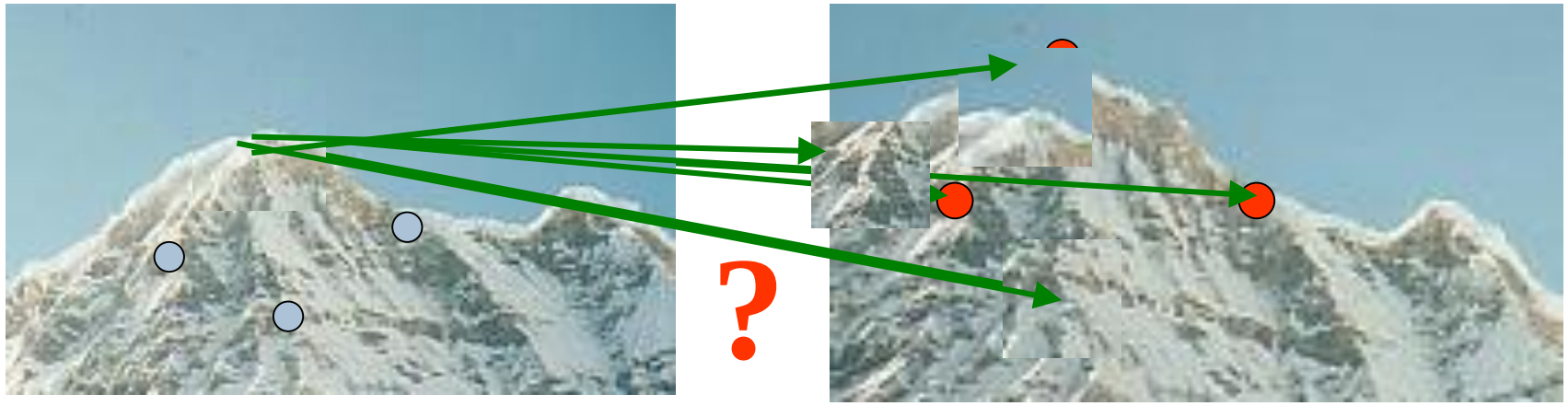


No chance to find true matches!

- Want to detect (at least some of) the same points in both images.
- Yet we have to be able to run the detection procedure *independently* per image.

Slide by James Hays

Goal: distinctive descriptor



- Want to be able to reliably determine which point goes with which.
- Must provide some invariance to geometric and photometric differences between the two views.

Slide by James Hays

Scale space

σ=0

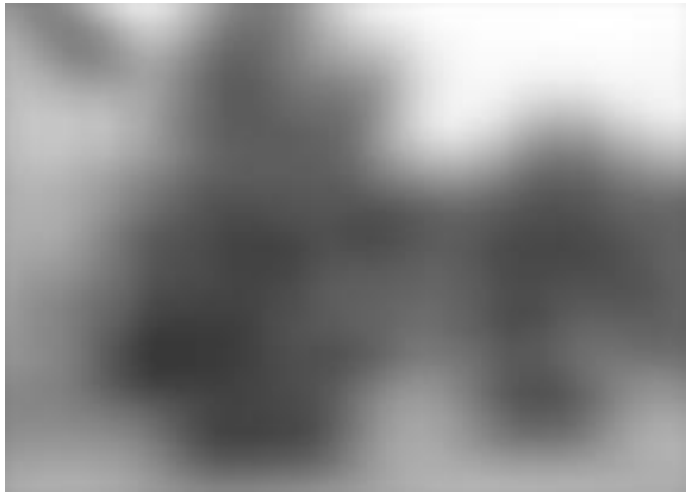
σ=1

σ=2

σ=4

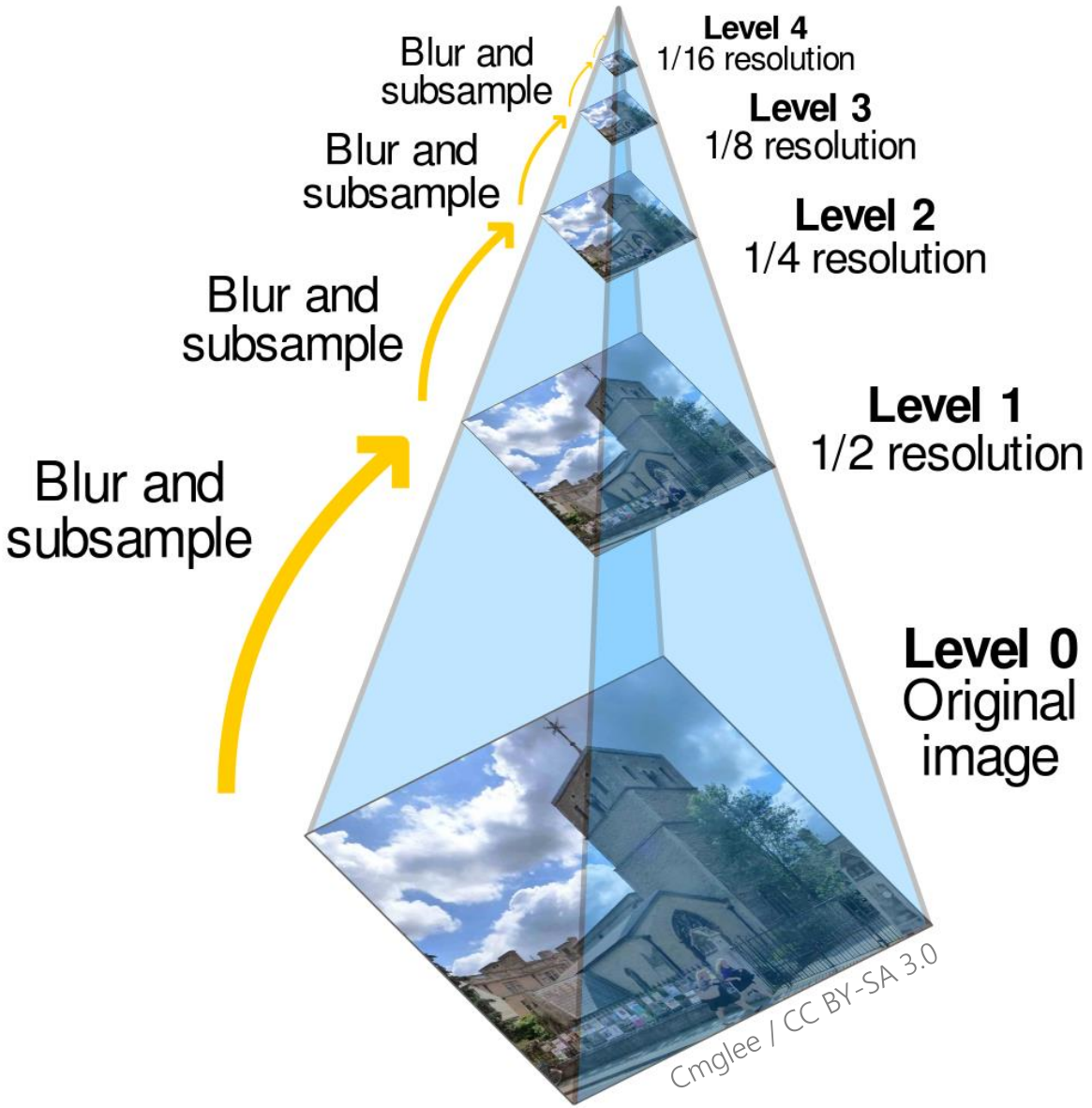
σ=8

σ=16

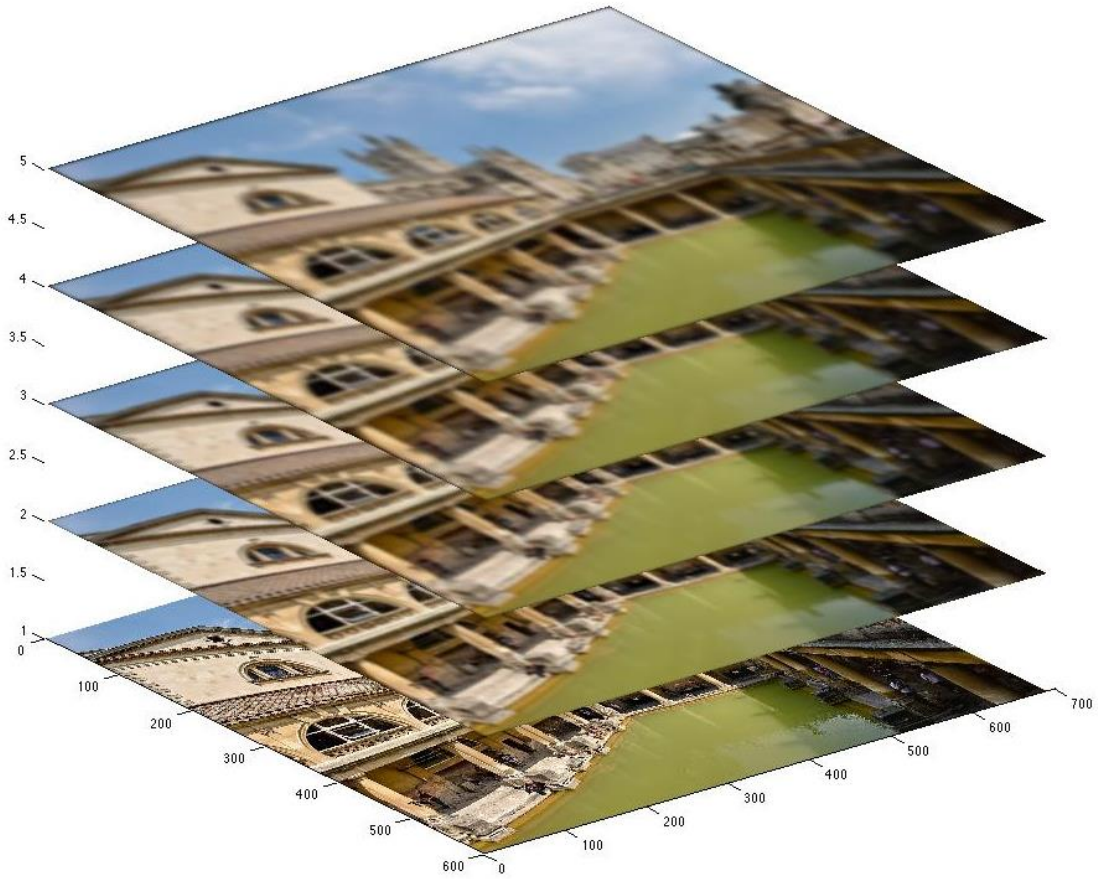


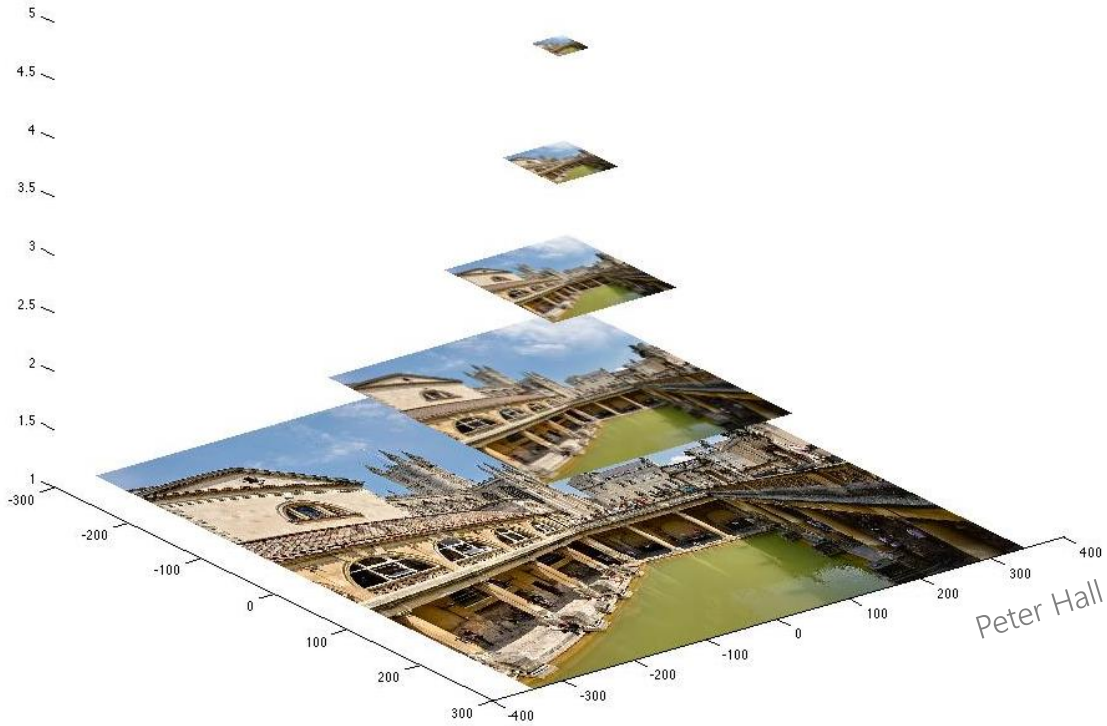
Pyramids

- Successive downsampling (blurring + subsampling) by a fixed factor:
  - ×2: most common (see right)
  - ×√2: two levels per octave (×2)
- Used in multi-resolution/ coarse-to-fine approaches
  - Solve a problem for small image at the top of the pyramid
  - Upsample solution to lower layer
  - Refine solution and continue



Scale space and pyramids

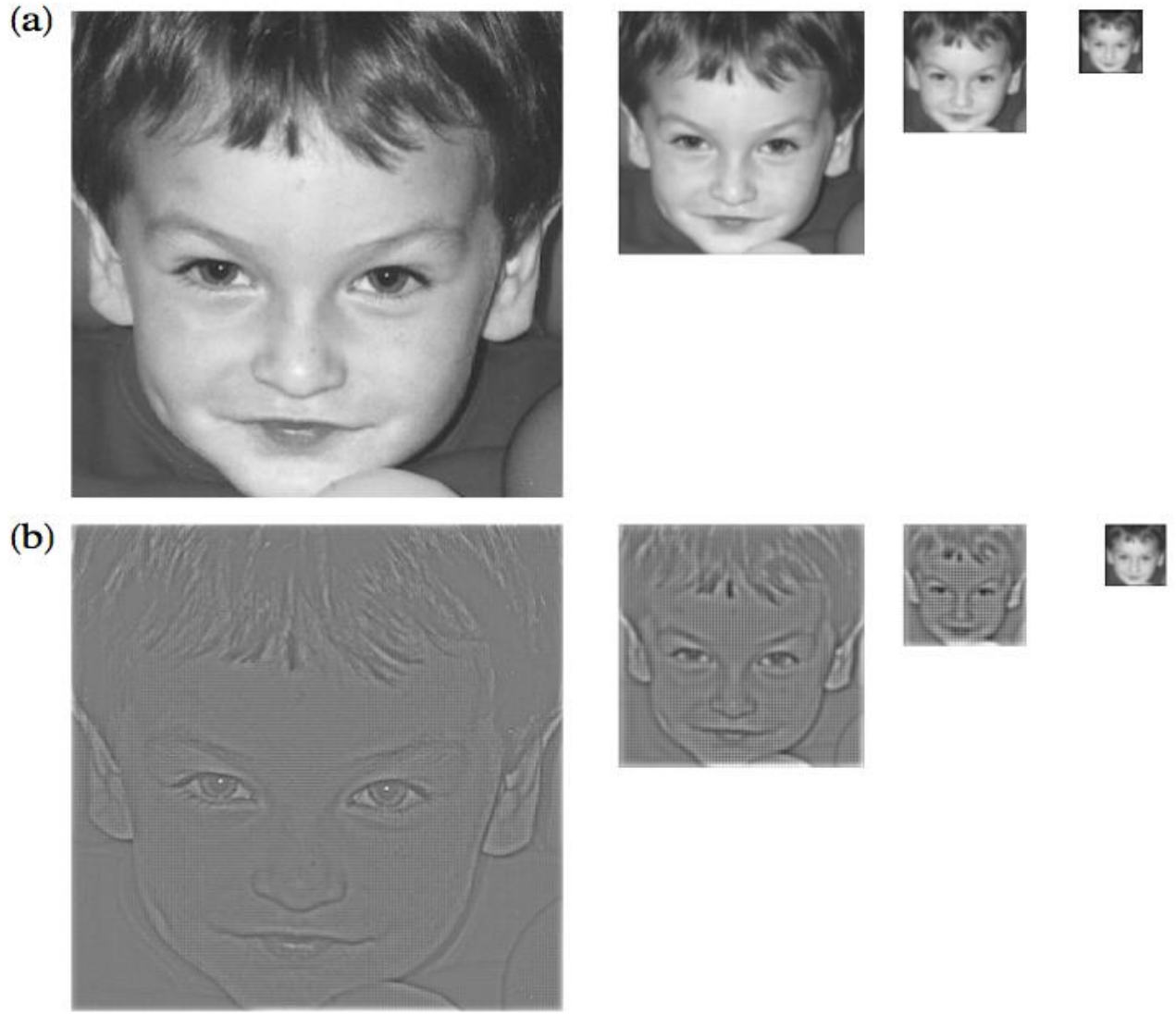




Gaussian and Laplacian pyramids

Gaussian pyramid

Laplacian pyramid



Brian A. Wandell, Foundations of Vision

Detecting edges (Canny)

Edges from gradients

gradient vector: $\nabla f = \frac{\partial f}{\partial x}\hat{x} + \frac{\partial f}{\partial y}\hat{y}$

gradient in *x*-direction:

$$\frac{\partial f}{\partial x} = \begin{bmatrix} -1/2 & 0 & +1/2 \end{bmatrix} * f$$

gradient in *y*-direction:

$$\frac{\partial f}{\partial y} = \begin{bmatrix} -1/2 \\ 0 \\ +1/2 \end{bmatrix} * f$$

gradient magnitude:

$$|\nabla f| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

gradient orientation:

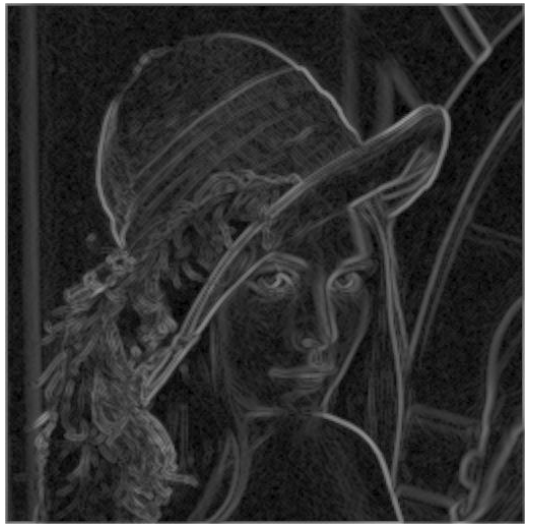
$$\theta = \mathrm{atan2}\left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$$

Edges from gradients

f

$\frac{\partial f}{\partial x}$

$\frac{\partial f}{\partial y}$



$|\nabla f|$

Note: images on the right enhanced to show more detail

Edges from gradients

Threshold to get a binary image:

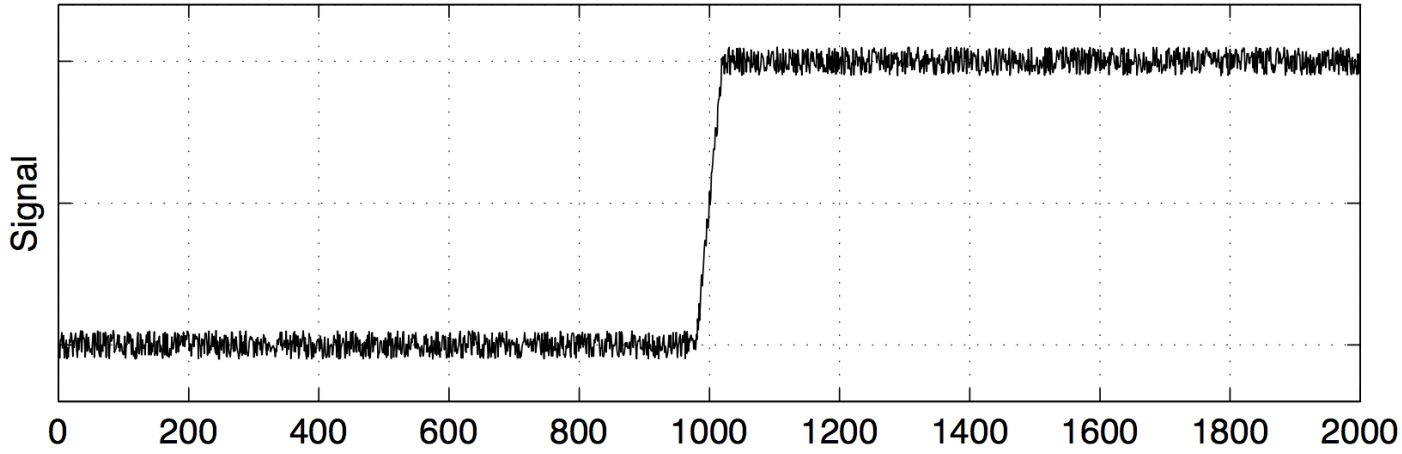
$|\nabla f| > 0.02$

$|\nabla f| > 0.05$

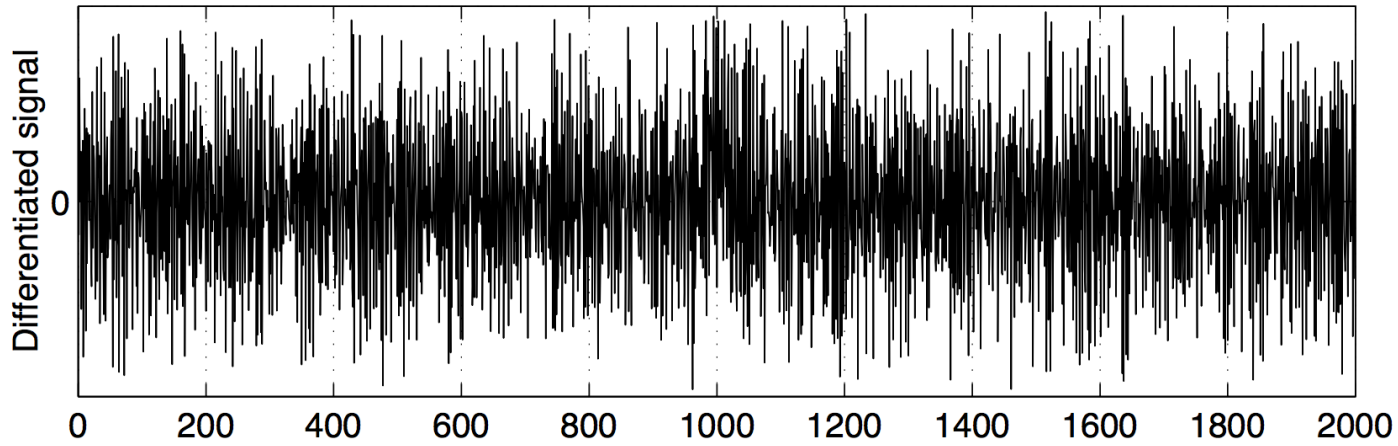
$|\nabla f| > 0.1$

Noisy signals

f

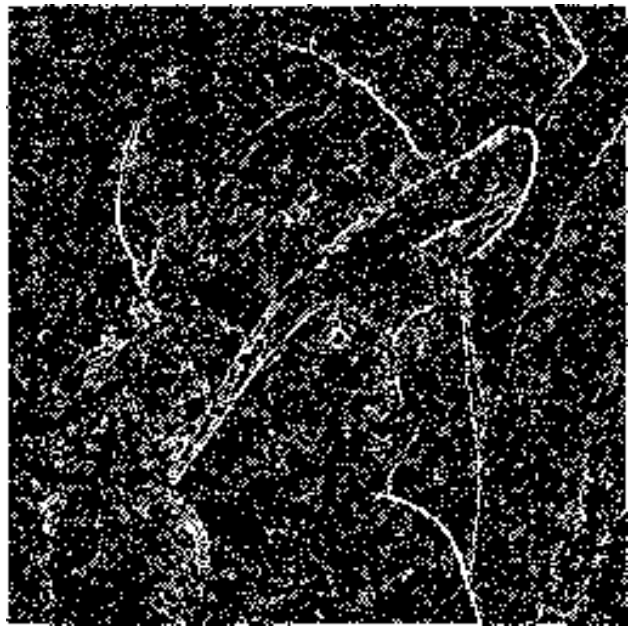


$\frac{df}{dx}$



Noisy images

Expect the same in images



Noisy images

- Solution: blur the image first

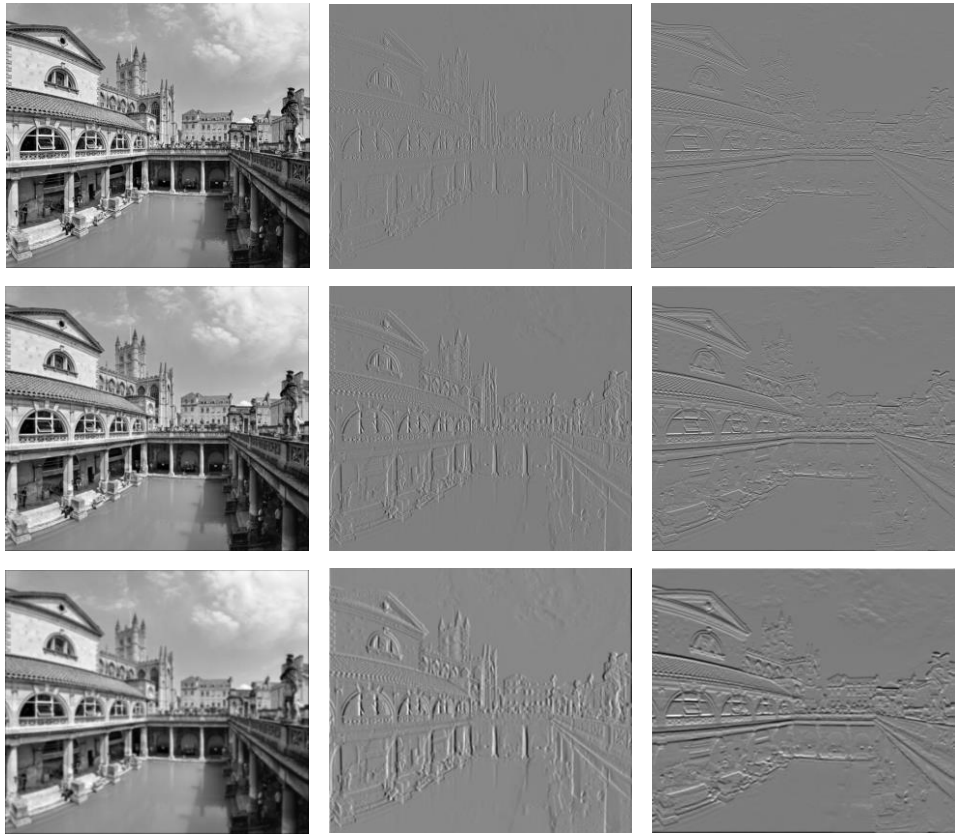
Notes on blurring

- The blur and derivative steps can be combined:

$$\begin{bmatrix} -1/2 & 0 & +1/2 \end{bmatrix} * (G * f) = (\begin{bmatrix} -1/2 & 0 & +1/2 \end{bmatrix} * G) * f$$

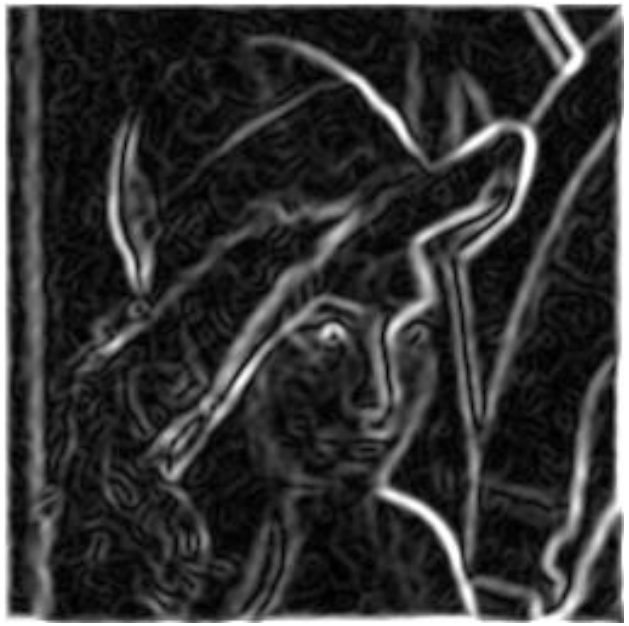
- Scale of the blur is important:
  The higher the blur from B,
  the more resilient to noise,
  but the more fine details lost

$$g(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

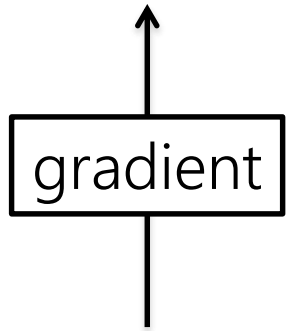


Edge thinning

- Non-maximum suppression for each of the gradient orientations



0	0	0	0	0
0.6	0.6	0.6	0.6	0.6
1	1	1	1	1
0.6	0.6	0.6	0.6	0.6
0	0	0	0	0



Gradient is north-south; so is the magnitude > than north-south neighbours?

Hysteresis thresholding

- Two threshold values: high (say 0.7) and low (say 0.5)

0	0.6	0	0	0
0	0	0	0	0
0.8	0.8	0.5	0.7	0.8
0	0	0	0	0
0	0	0	0	0

As the high threshold is 0.7, the pixel outlined in red will not be considered an edge pixel in the first pass. However, if the low threshold is 0.5 or lower, it will be accepted as it lies on a curve with other edge pixels.

Canny edge detector

1. Gaussian blur the input image
2. Compute gradient magnitude maps in 4 directions: — \ | /
3. Edge thinning:
    1. Non-maximum suppression along each gradient direction
4. Hysteresis thresholding:
    1. Identify strong edge pixels > high threshold
    2. Trace connected edge pixels > low threshold
5. Output binary map of Canny edges

J. Canny. *A Computational Approach To Edge Detection*.
IEEE Transactions on Pattern Analysis and Machine Intelligence, 8(6):679–698, 1986.

Canny edges

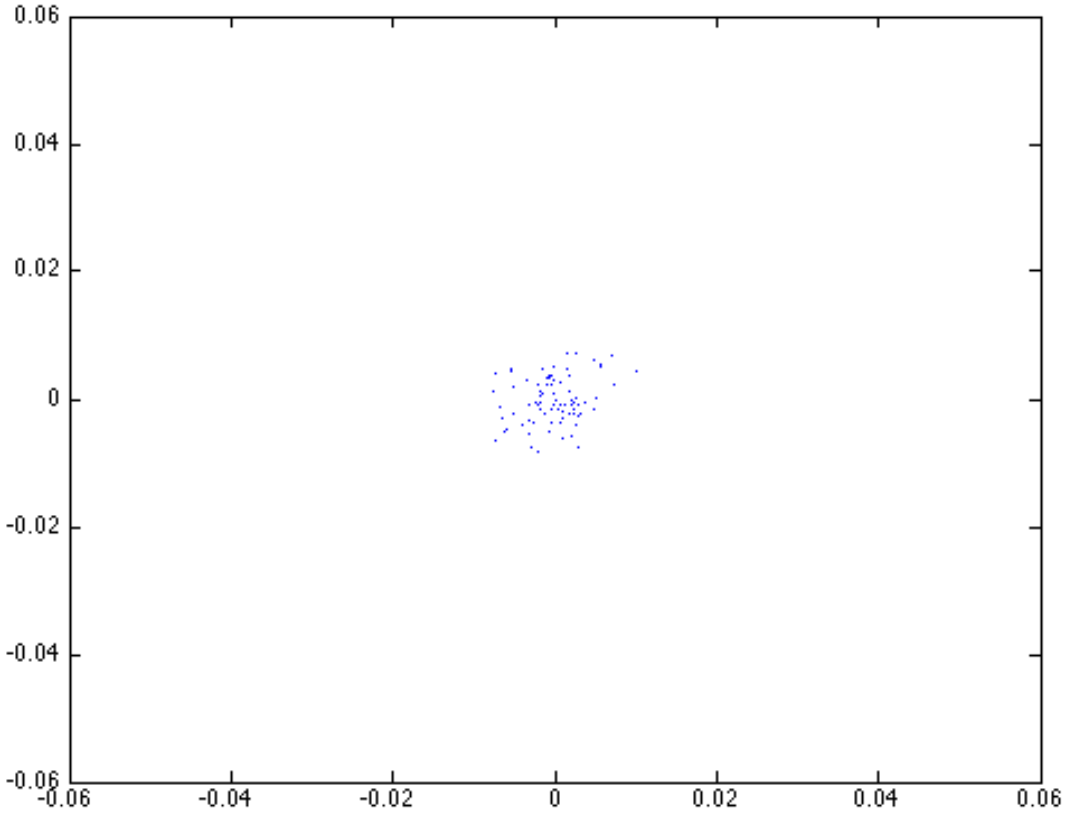
Canny edges

JonMcLoone at English Wikipedia / CC BY-SA 3.0

Detecting corners (Harris)

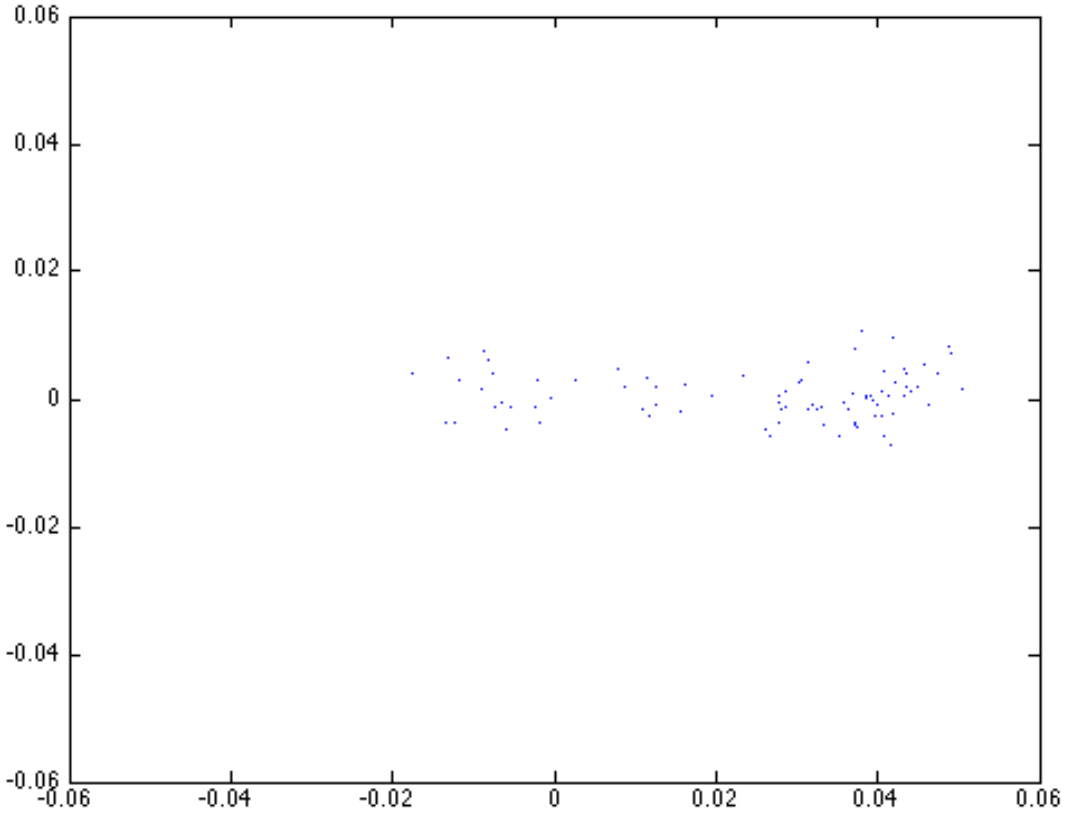
Harris corners

- In a patch, plot $\frac{\partial f}{\partial x}$ against $\frac{\partial f}{\partial y}$



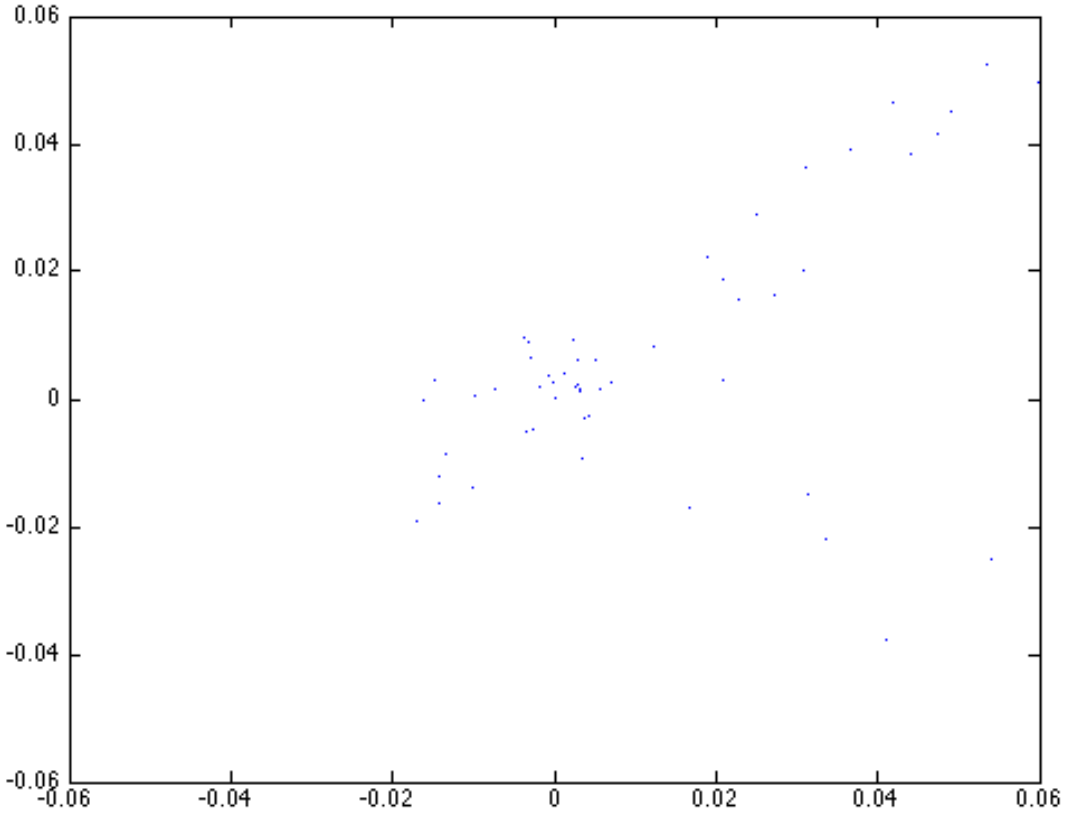
Harris corners

- In a patch, plot $\frac{\partial f}{\partial x}$ against $\frac{\partial f}{\partial y}$



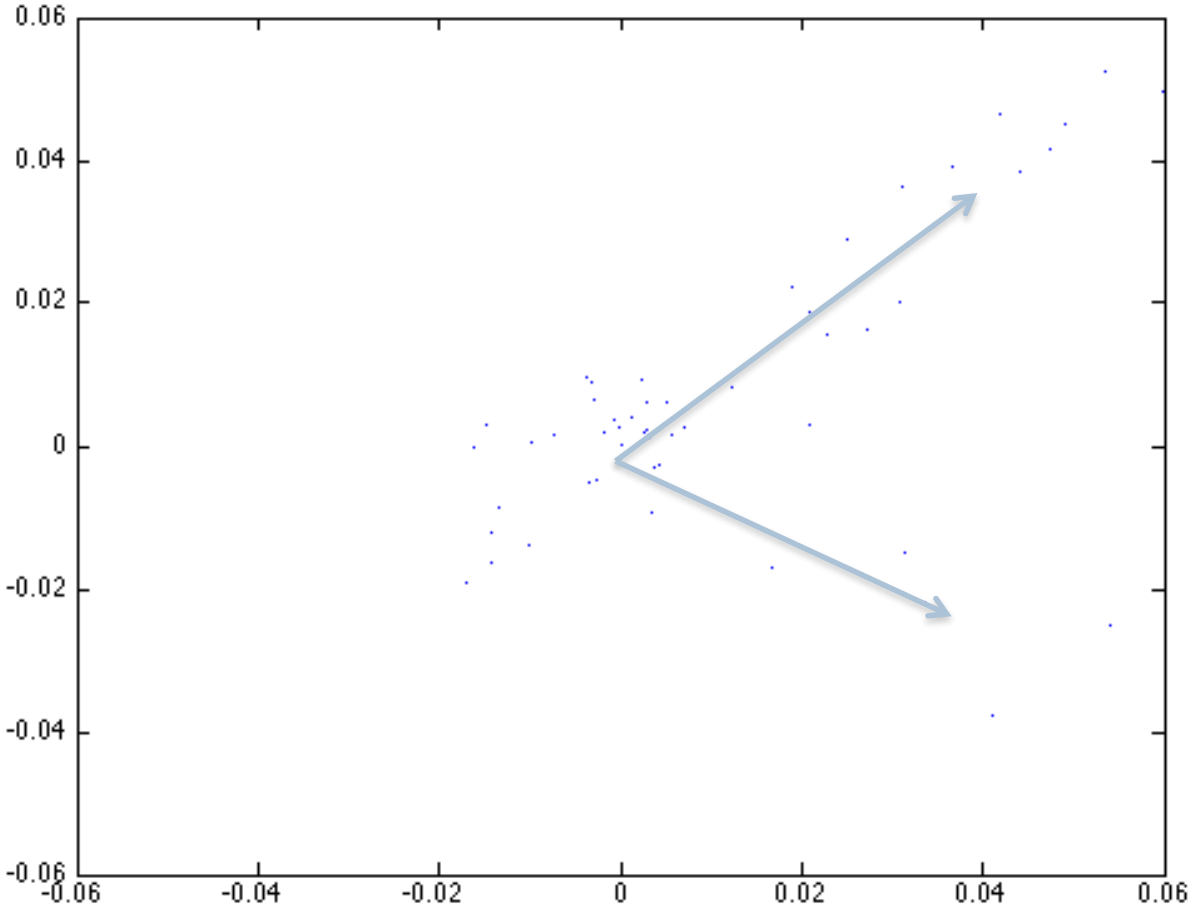
Harris corners

- In a patch, plot $\frac{\partial f}{\partial x}$ against $\frac{\partial f}{\partial y}$

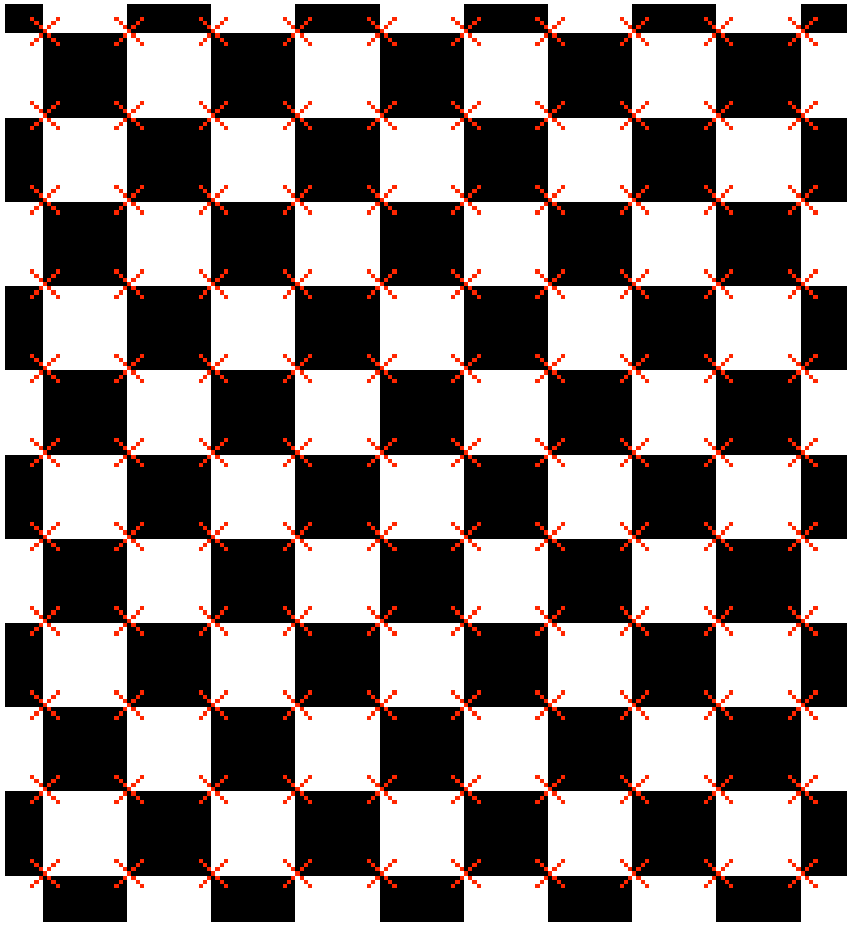


Harris corners

- Calculate eigenvalues of covariance matrix:
  - Two small eigenvalues: no features of interest, i.e. mostly uniform intensity
  - One small and one large eigenvalue: found an edge
  - Two large eigenvalues: found a corner



Harris corners



Harris corners

Harris corner detection

1. Compute gradient maps I_x and I_y
2. Compute structure tensor:

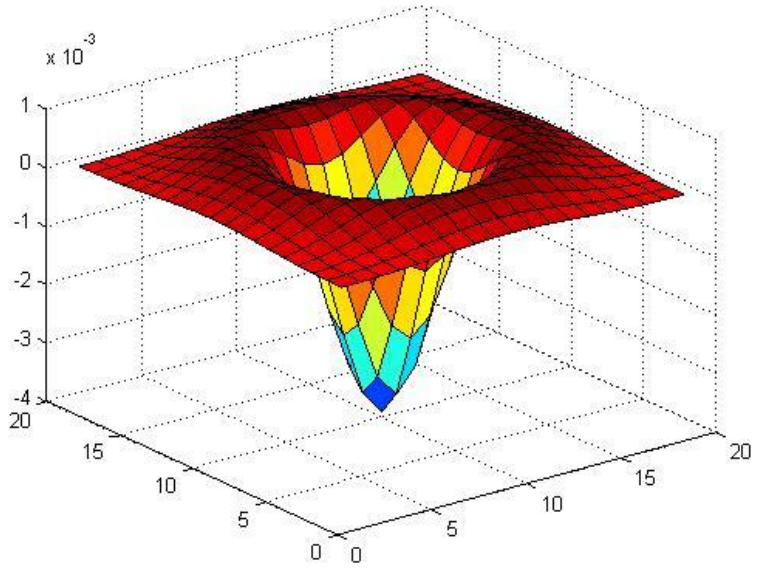
$$A = \sum_u \sum_v w(u,v) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

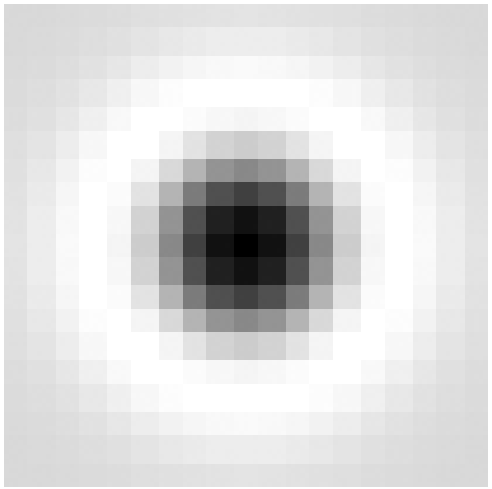
3. Compute 'cornerness score': $M_c = \det(A) - k \cdot \mathrm{Tr}^2(A)$
4. Find local maxima of 'cornerness score':
    1. Maximum values among surrounding pixels (8-neighbours)
5. Output binary map of corners

C. Harris and M. Stephens. *A combined corner and edge detector.*
Proceedings of the 4th Alvey Vision Conference, 1988, pp 147–151.

Detecting blobs (DoG)

Laplacian of Gaussian (LoG)





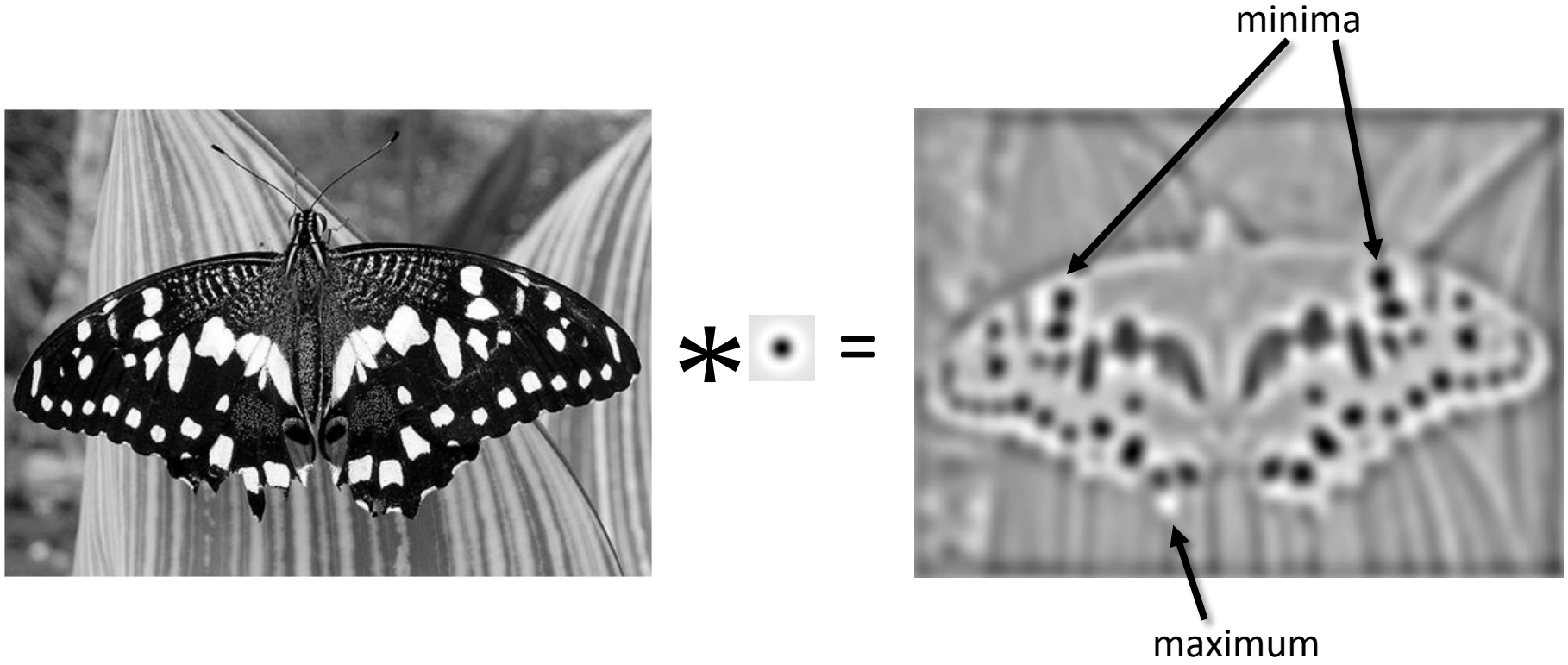
$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

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

Slide by Noah Snavely

Laplacian of Gaussian

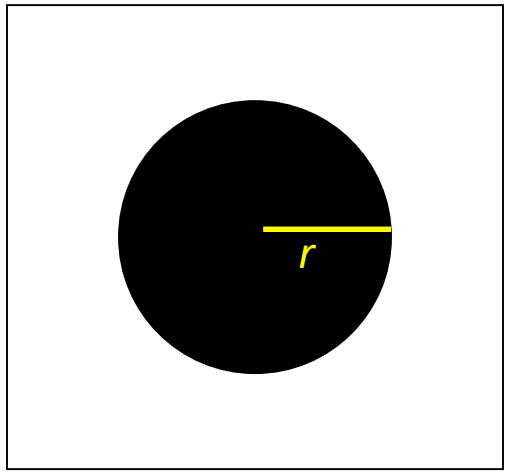
- "Blob" detector – finding minima and maxima in space and scale



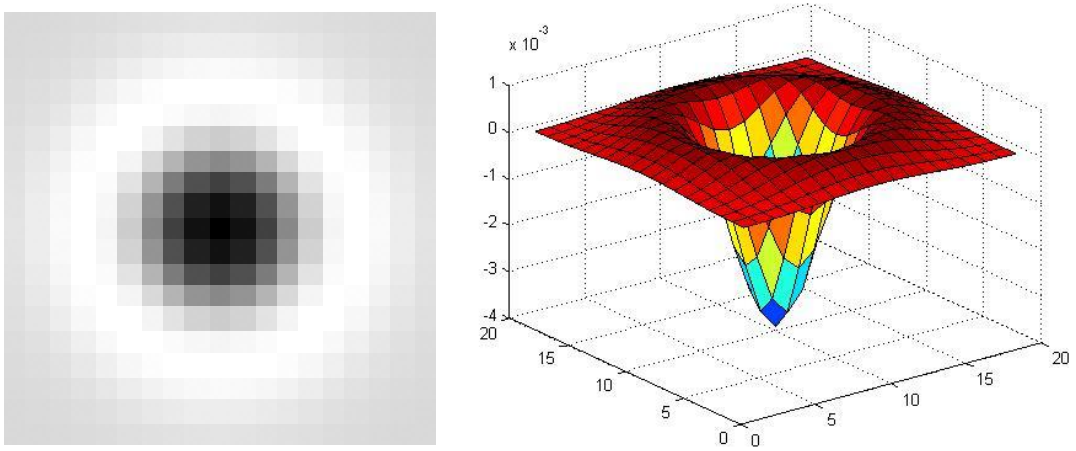
Slide by Noah Snavely

Scale selection

- At what scale does the Laplacian achieve a maximum response for a binary circle of radius *r*?



Image

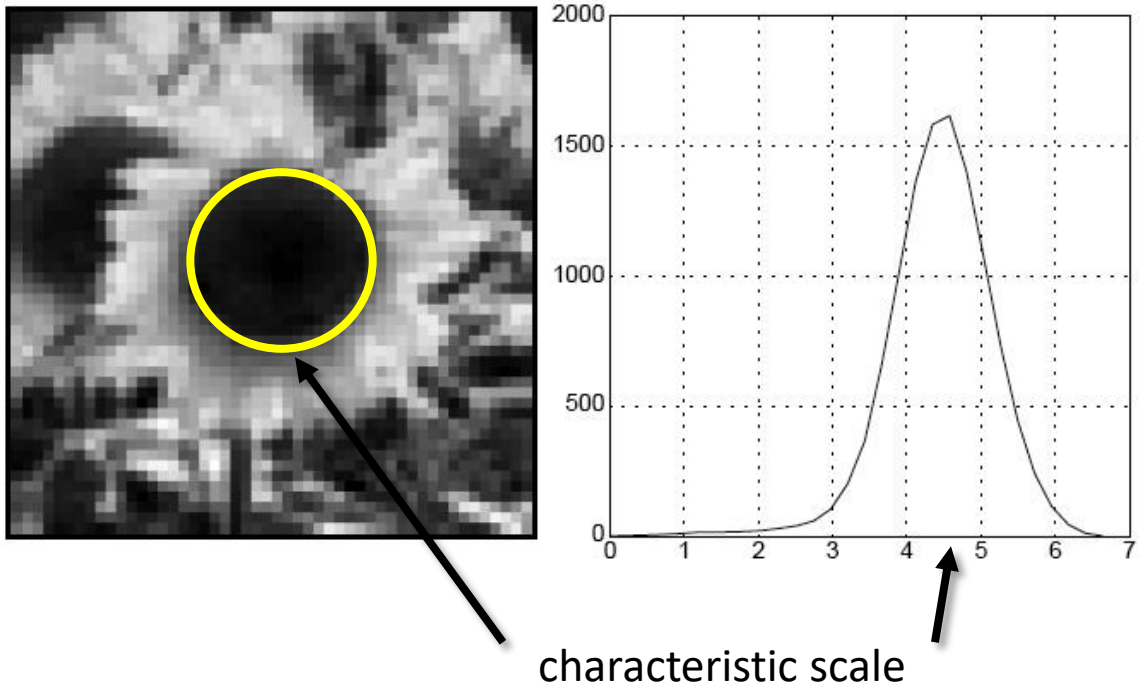


Laplacian

Slide by Noah Snavely

Characteristic scale

- Defined as the scale that produces the peak of Laplacian response



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

Slide by Noah Snavely

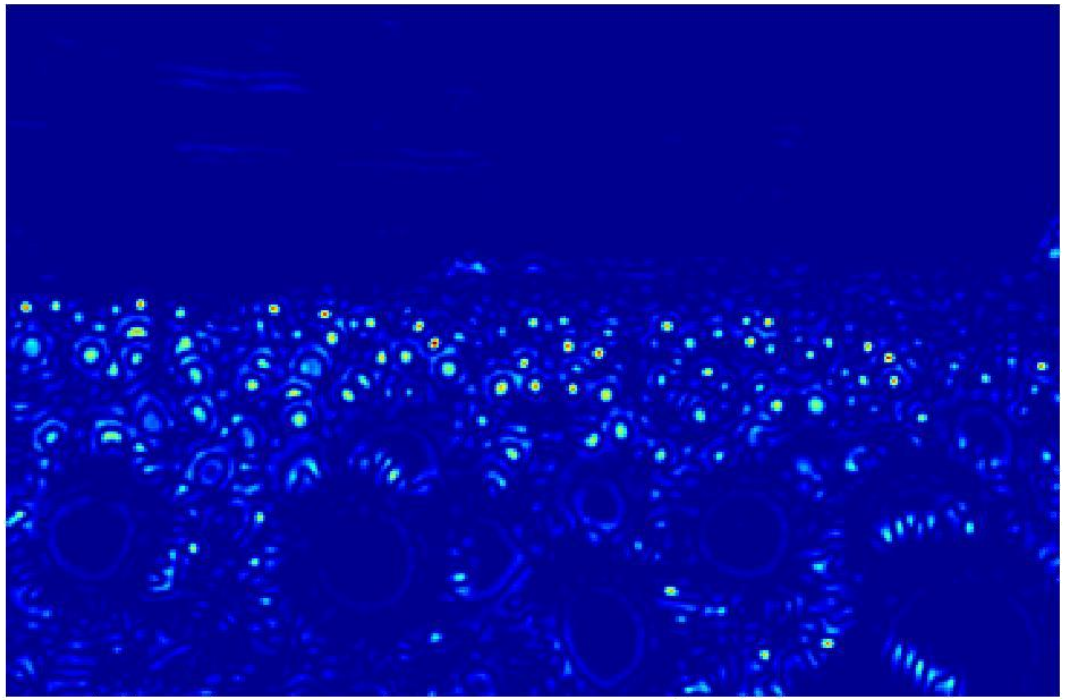
Scale-space blob detector: Example

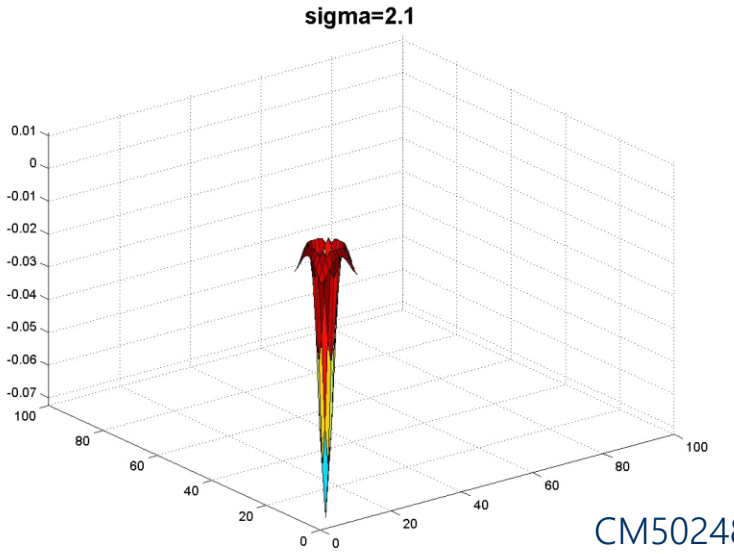
Original image at ¾ the size

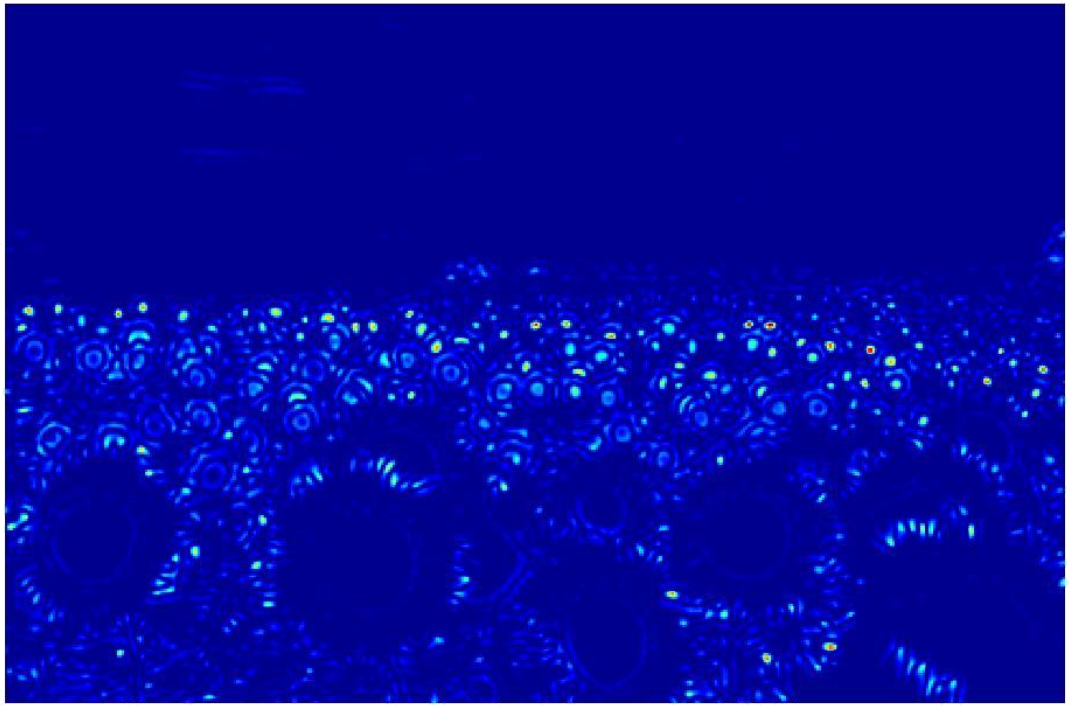
Slide by Kristen Grauman

Scale-space blob detector: Example

Original image at ¾ the size

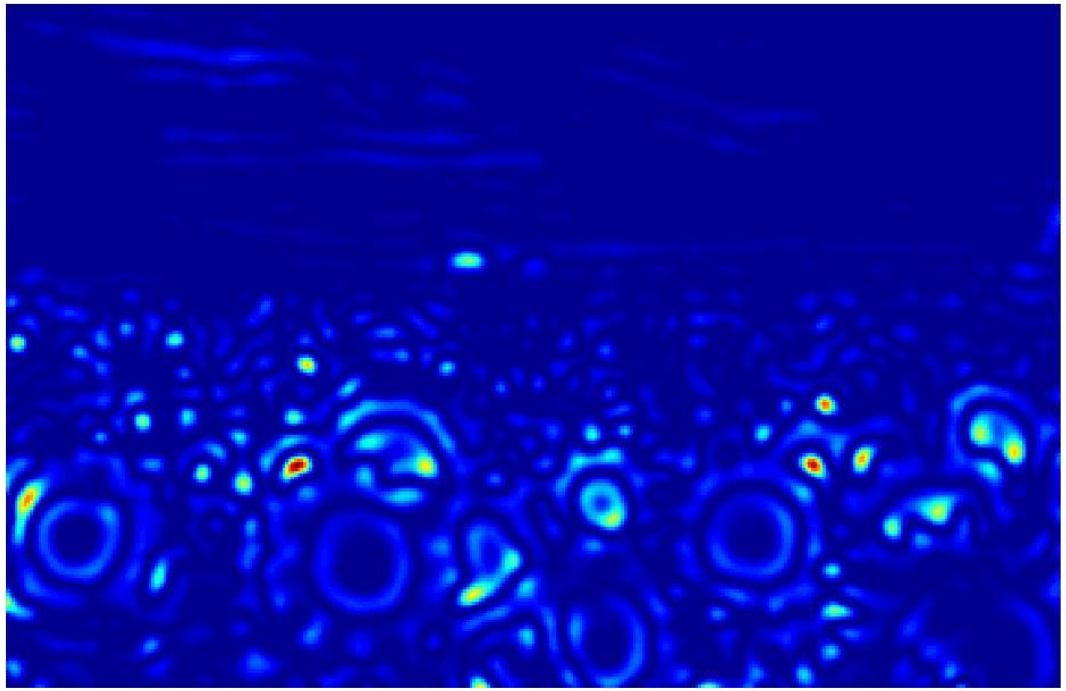


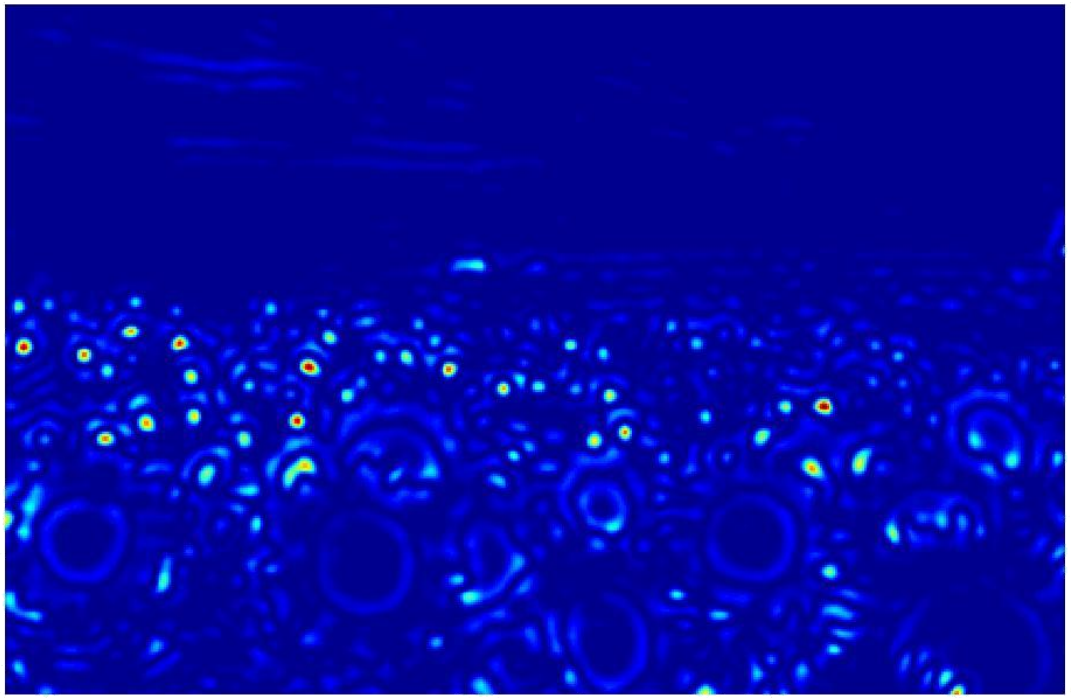


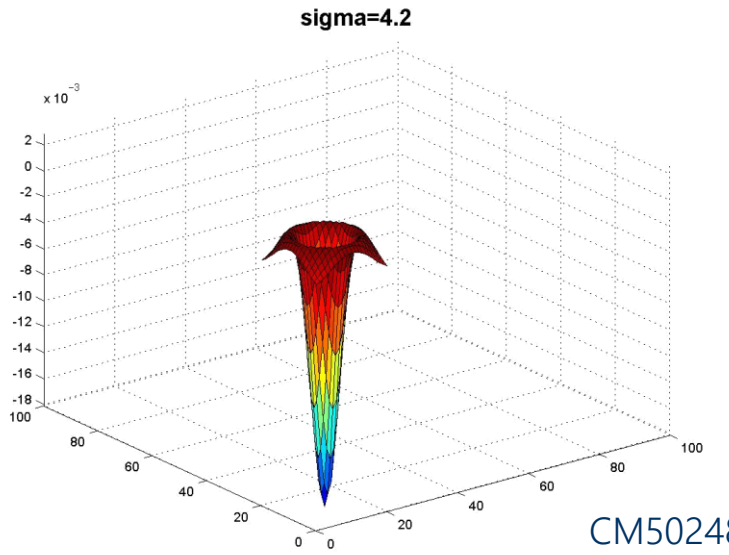


Slide by Kristen Grauman

Scale-space blob detector: Example

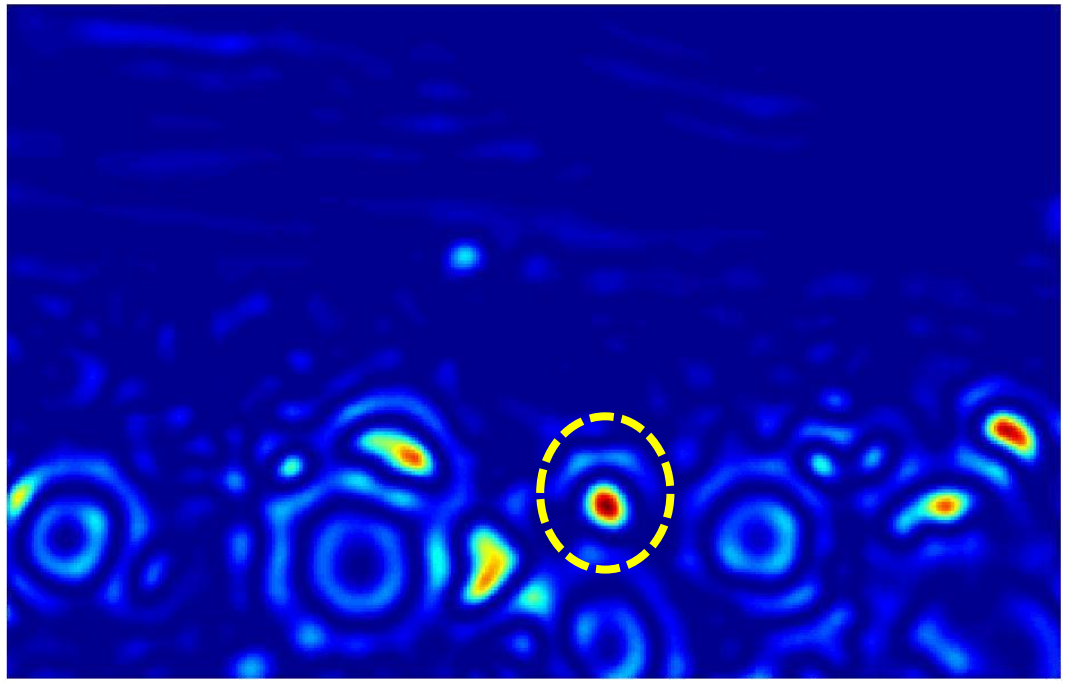


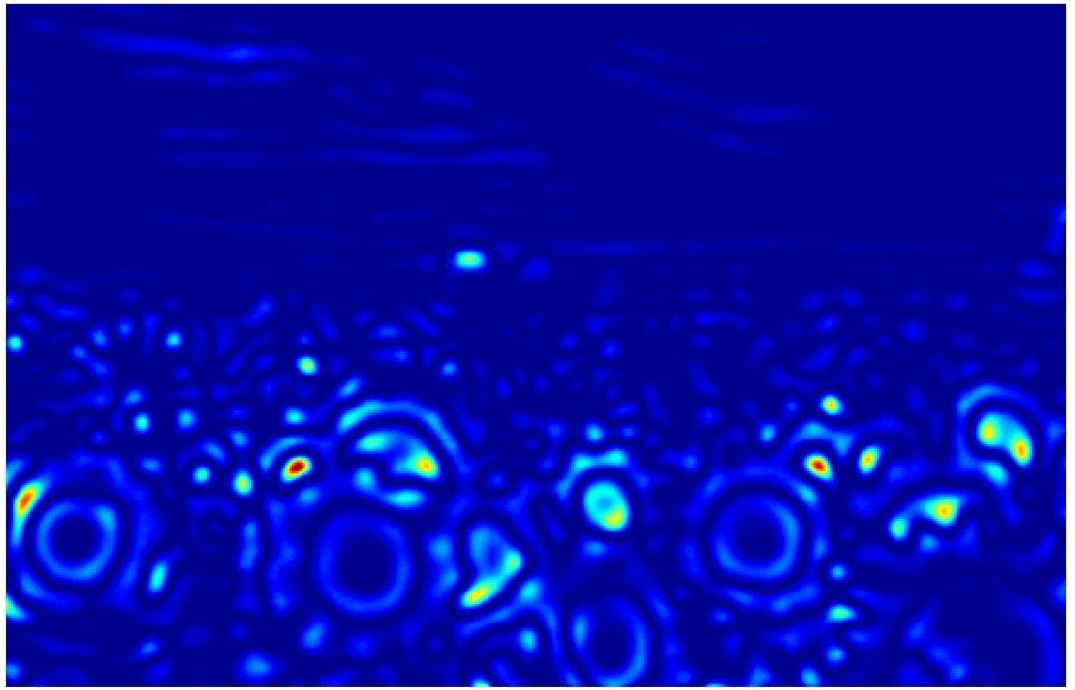


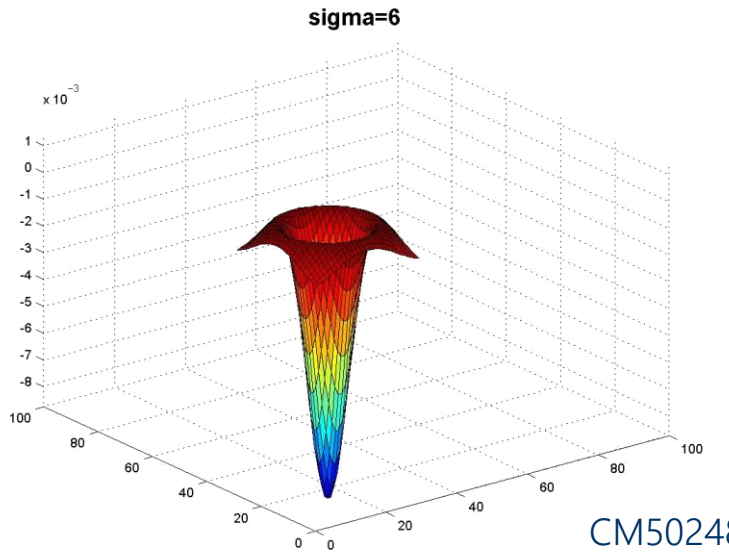


Slide by Kristen Grauman

Scale-space blob detector: Example

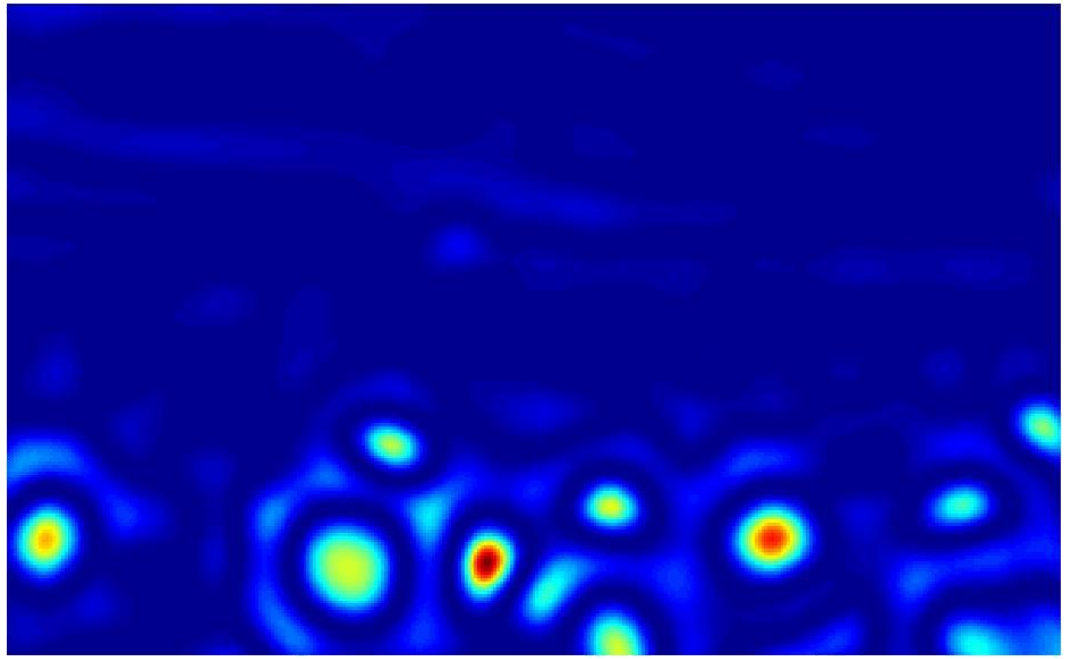


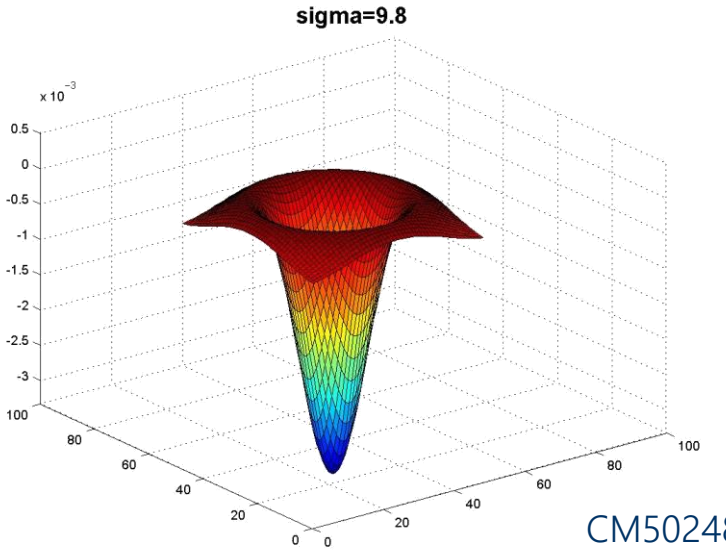


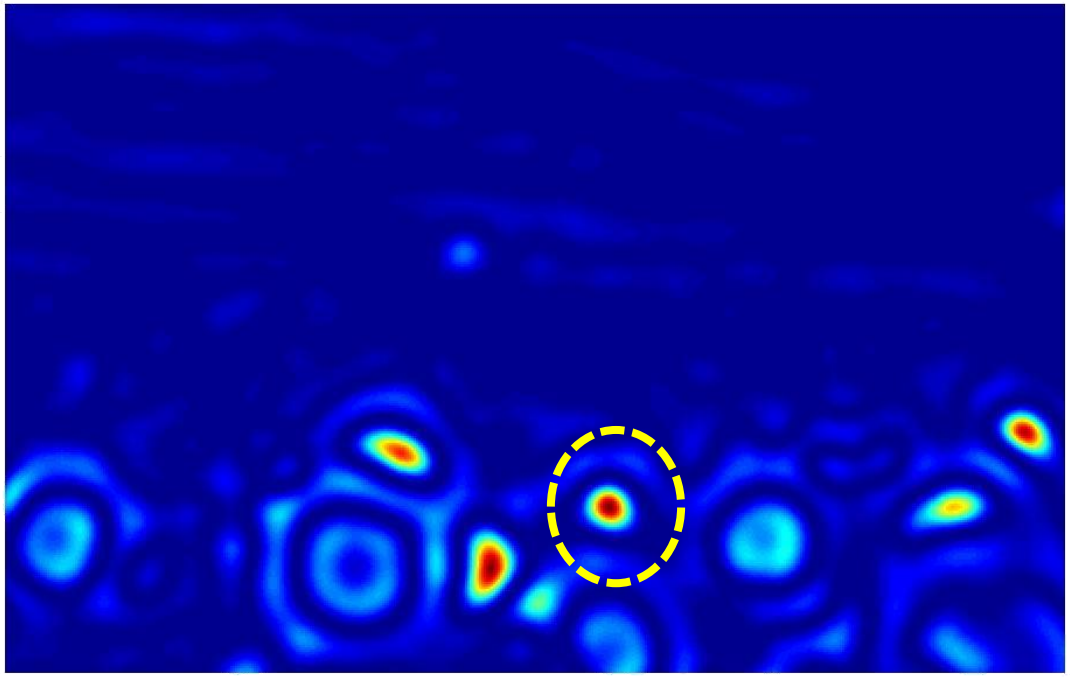


Slide by Kristen Grauman

Scale-space blob detector: Example

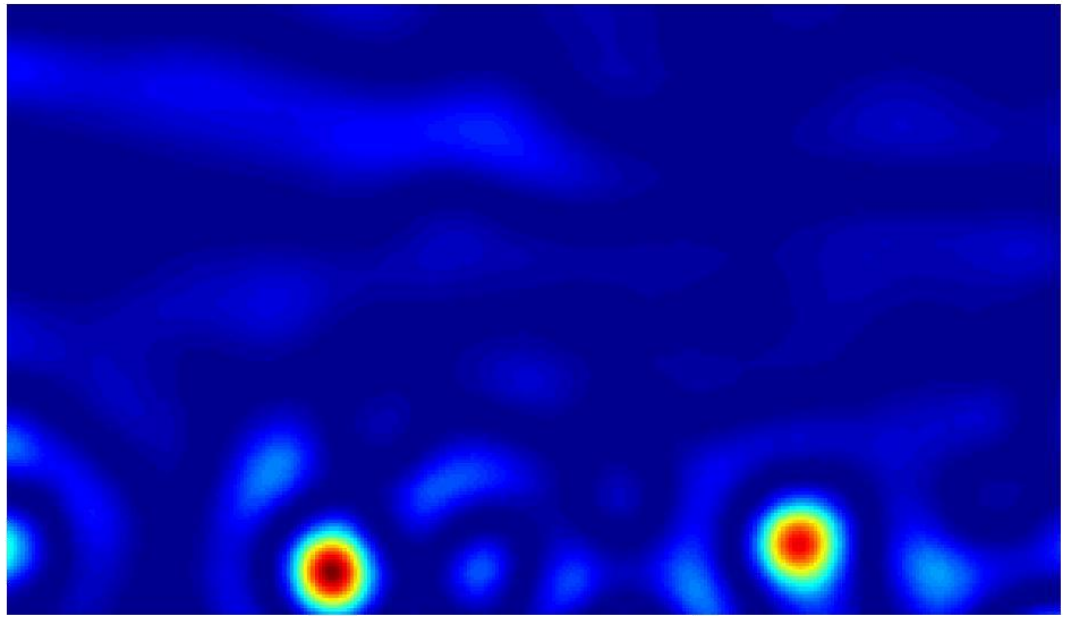


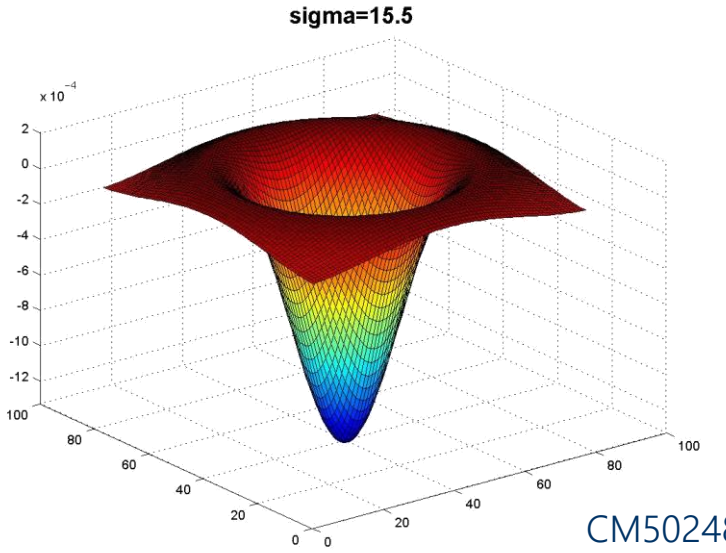


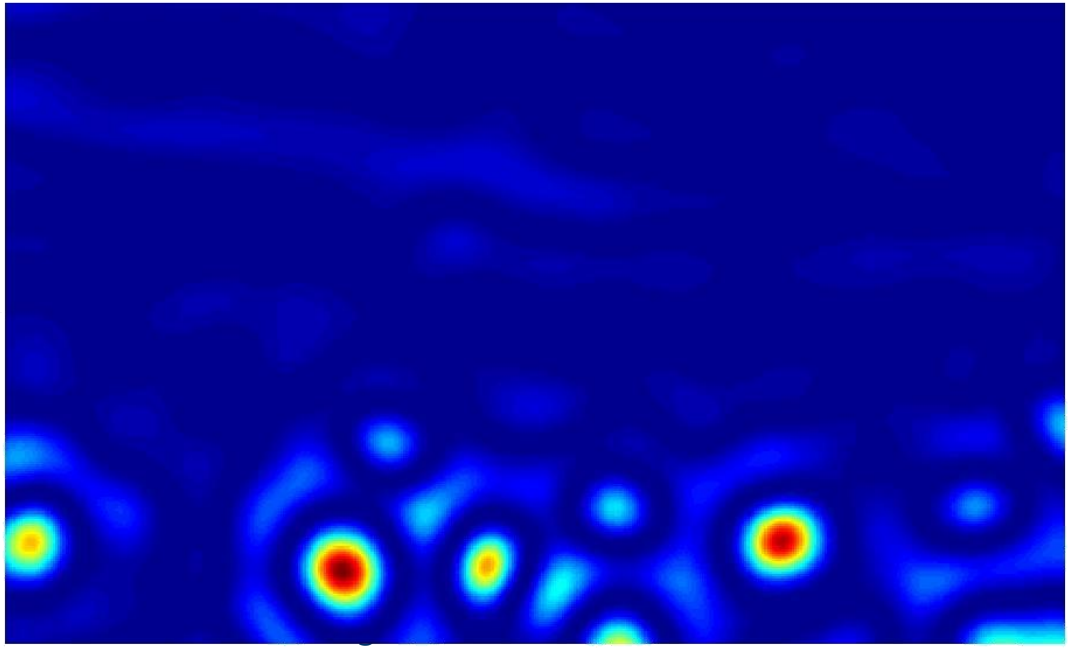


Slide by Kristen Grauman

Scale-space blob detector: Example



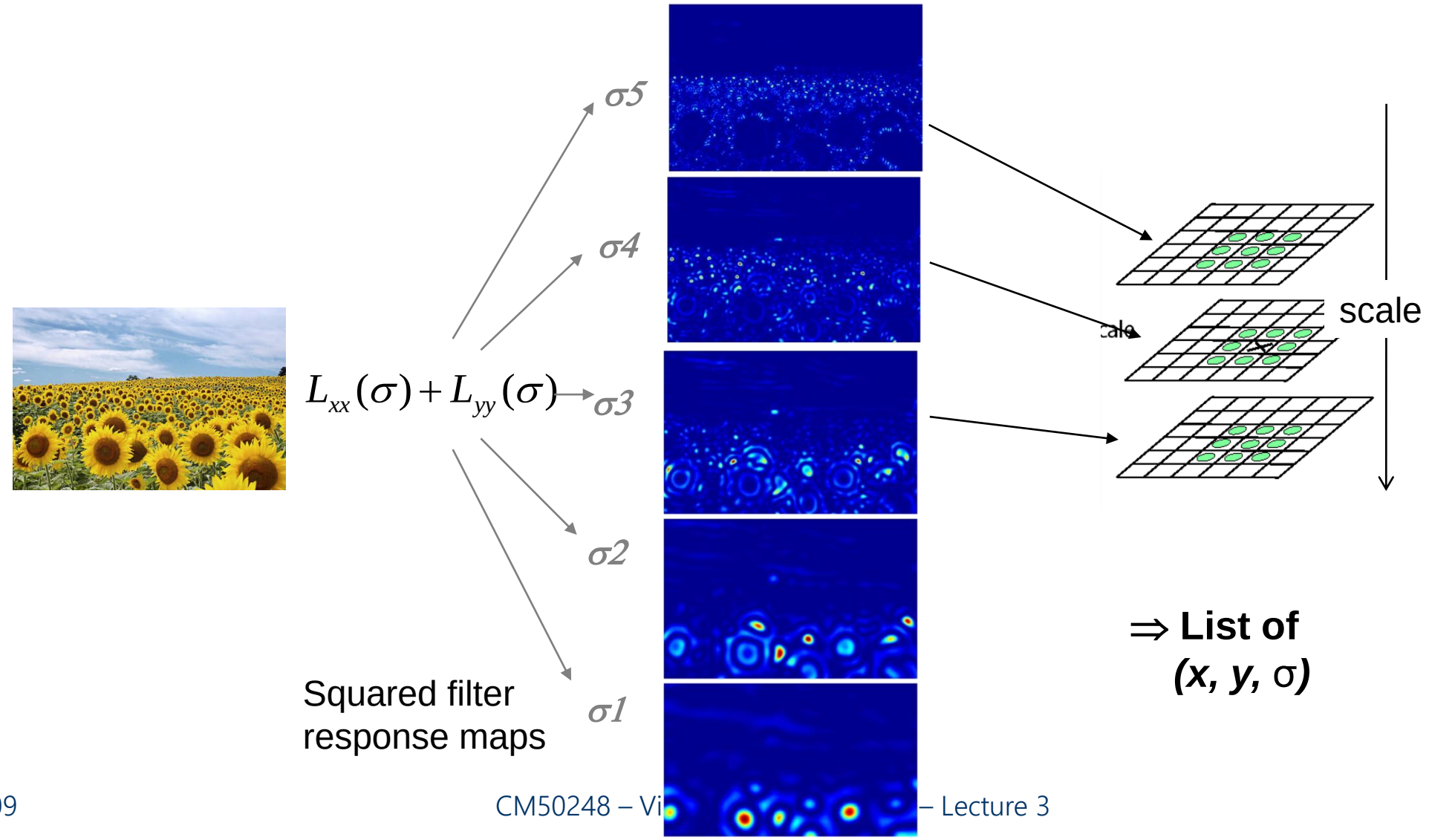




Slide by Kristen Grauman

Scale-invariant interest points

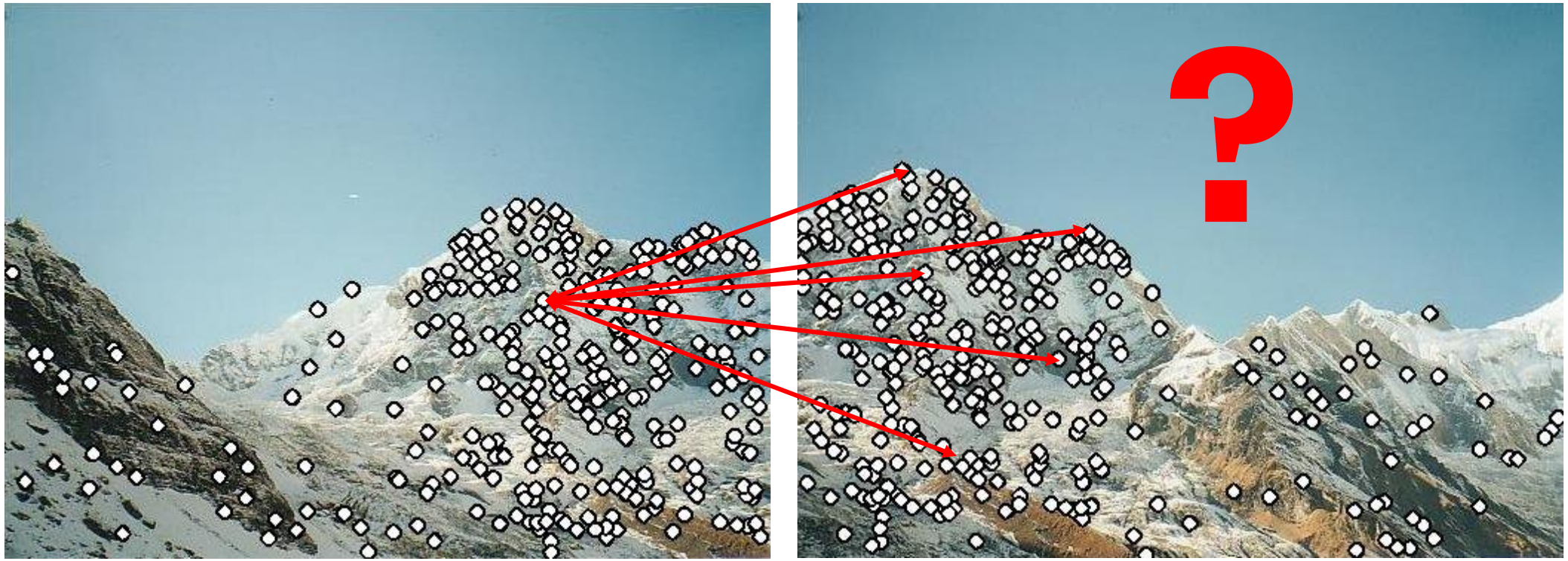
Interest points are local maxima in both position and scale.



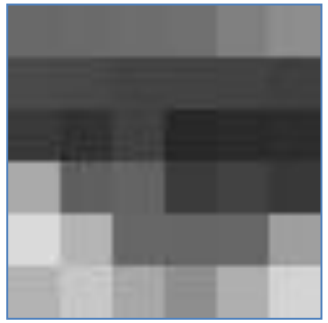
Slide by James Hays

Descriptors

Descriptors



Descriptors

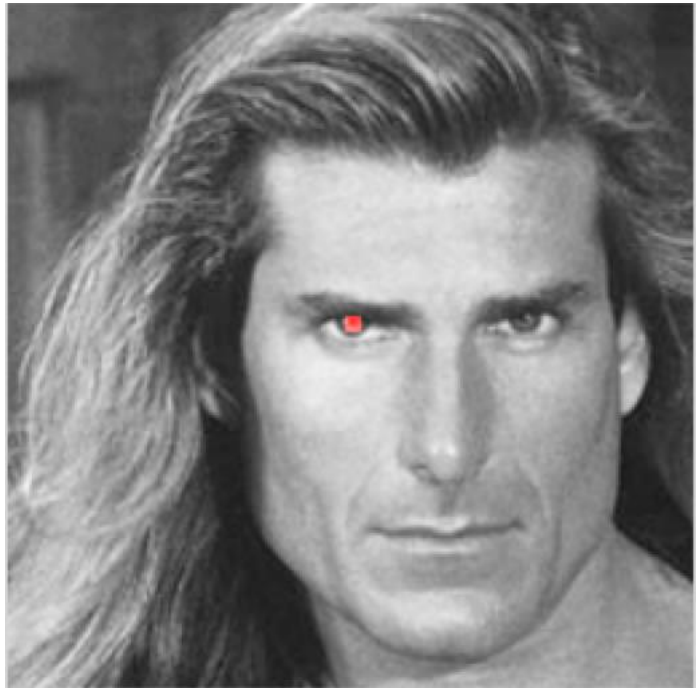


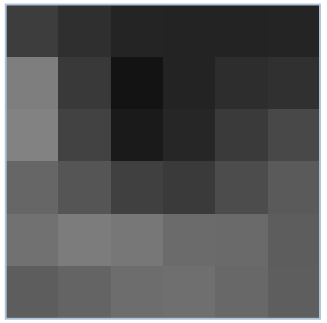
0.412	0.418	0.431	0.439	0.503	0.558
0.289	0.282	0.262	0.263	0.262	0.246
0.224	0.196	0.255	0.155	0.168	0.176
0.667	0.374	0.393	0.229	0.259	0.219
0.853	0.708	0.404	0.401	0.404	0.622
0.702	0.779	0.658	0.556	0.685	0.830

Compare two patches with, for example, sum squared differences

Descriptors

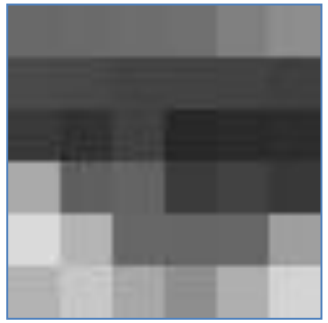
Possible issues: choice of descriptor

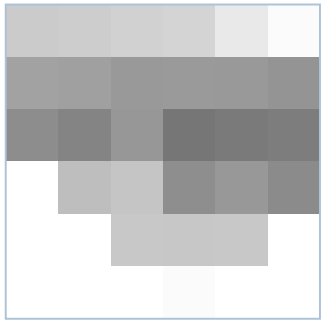




Descriptors

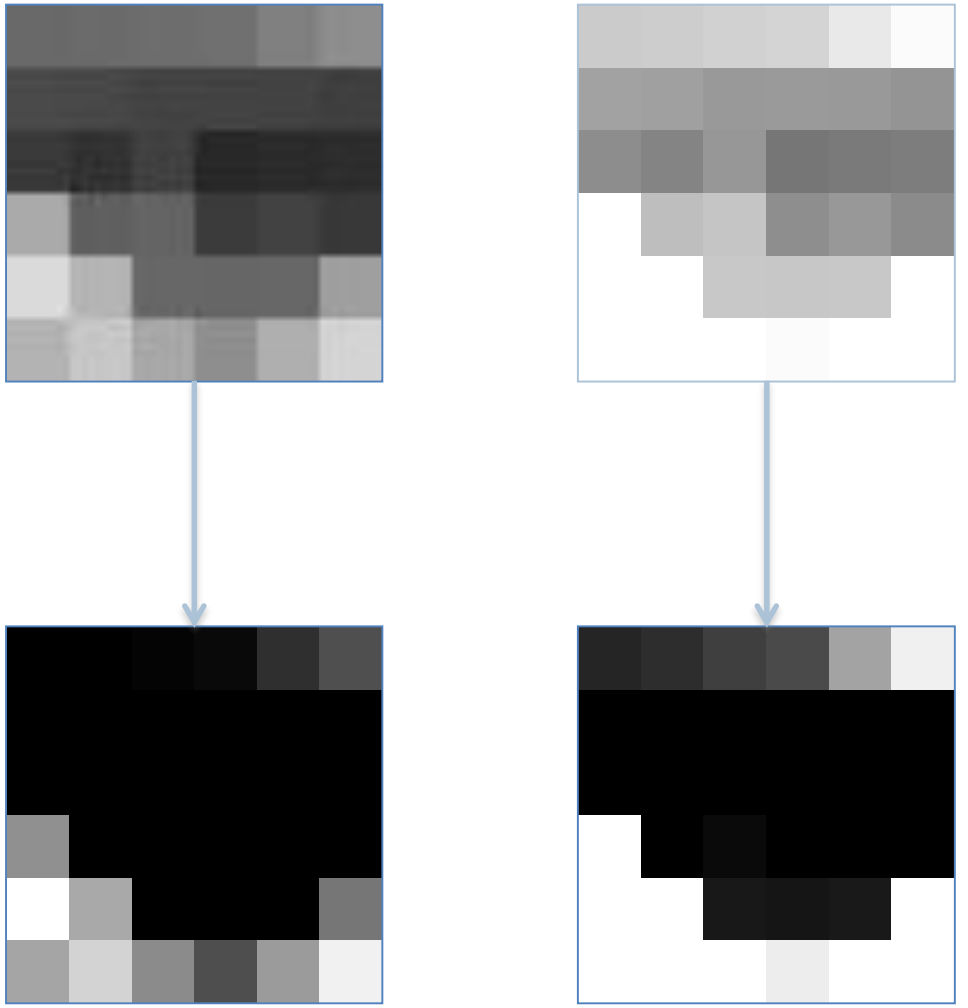
Possible issues: how we compare descriptors





Zero-mean normalisation

- Account for some changes in illumination:
  - From each pixel subtract the mean of pixels within the patch
  - Also divide by the standard deviation of pixels in the patch



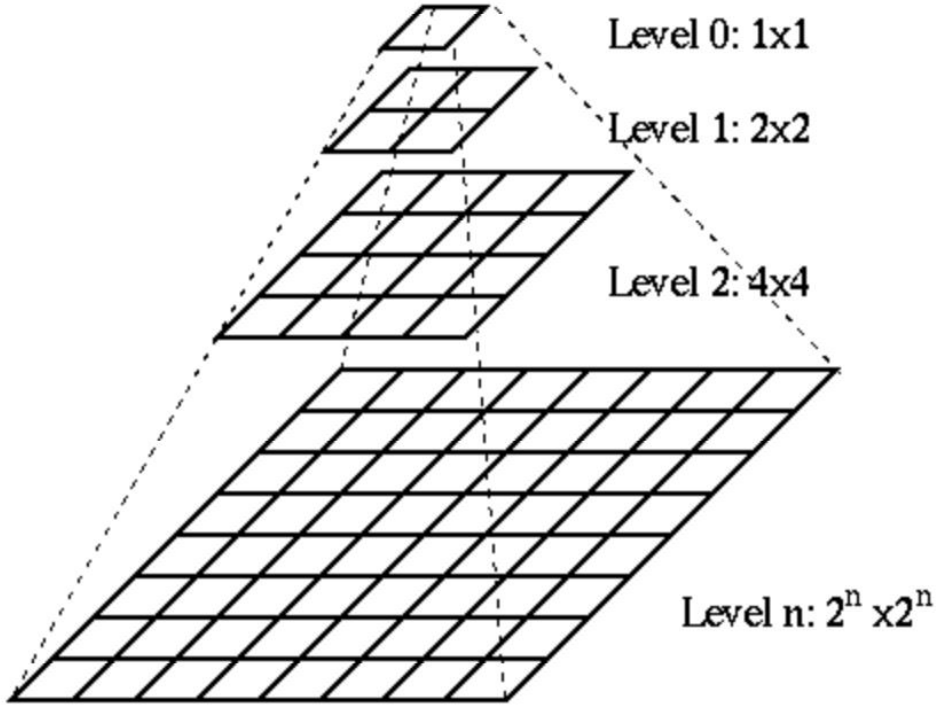
Invariance

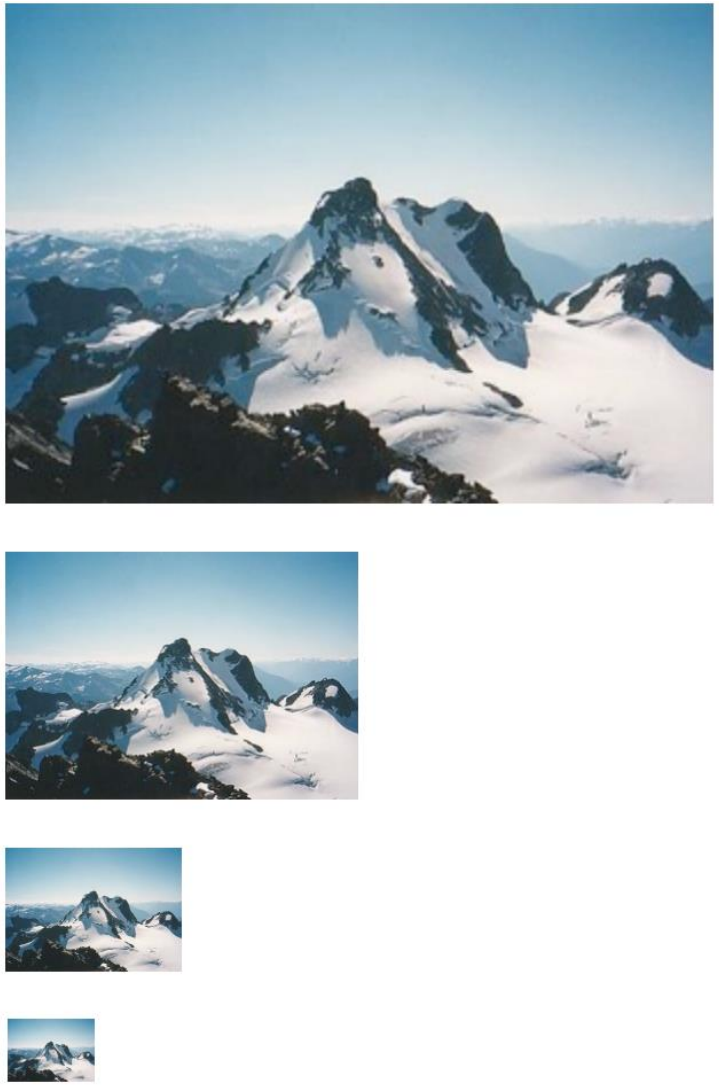
- Translation?
- Scale?
- Rotation?
- Projection?
- Illumination?

Original Translated

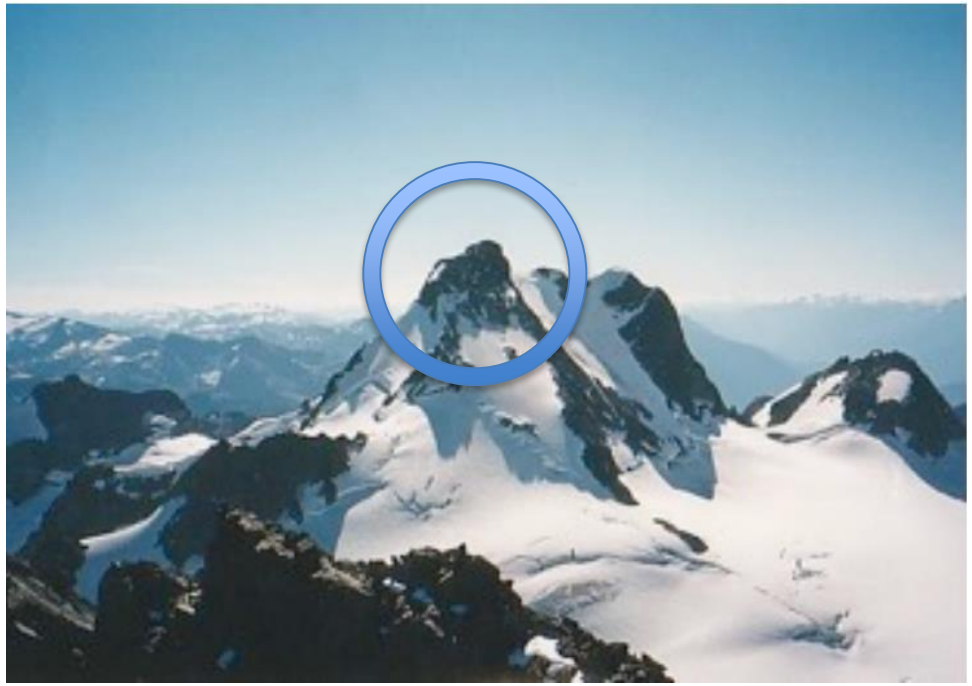
Rotated Scaled

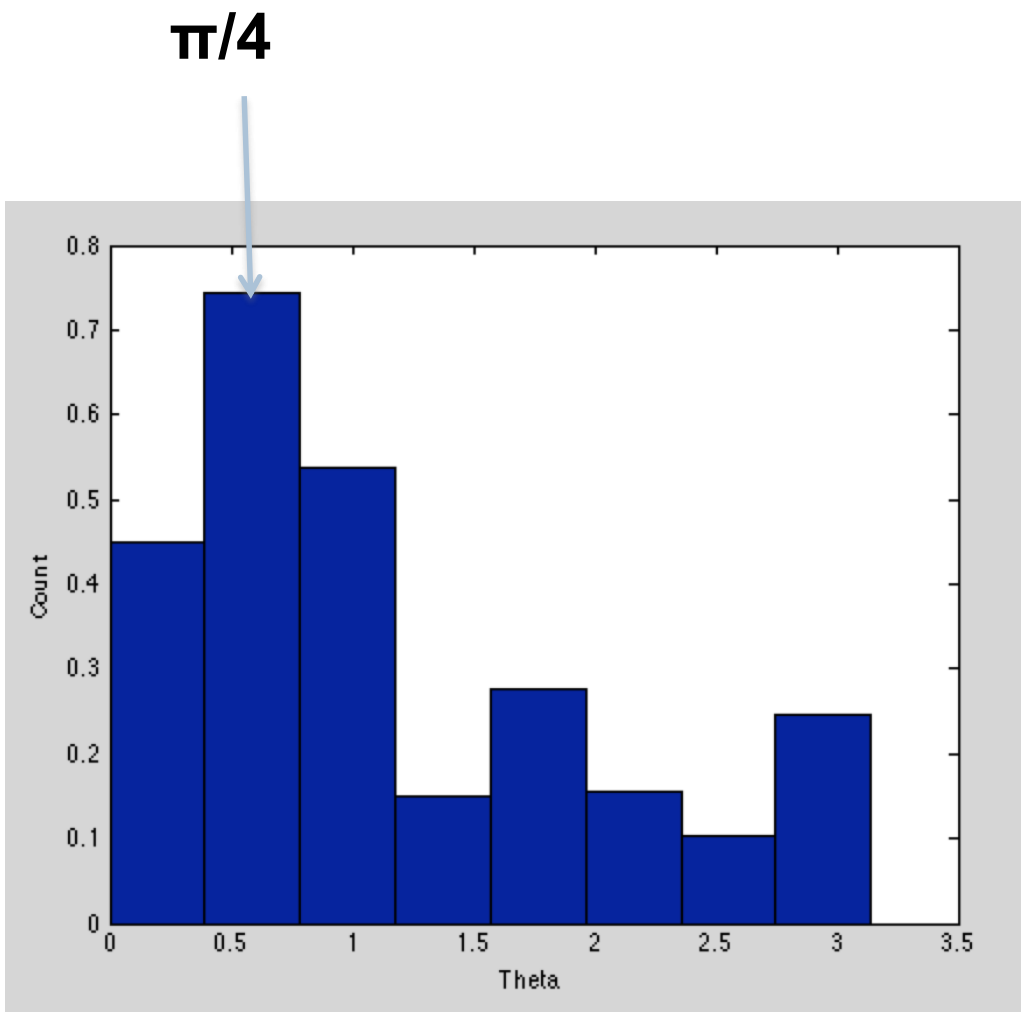
Invariance to scale





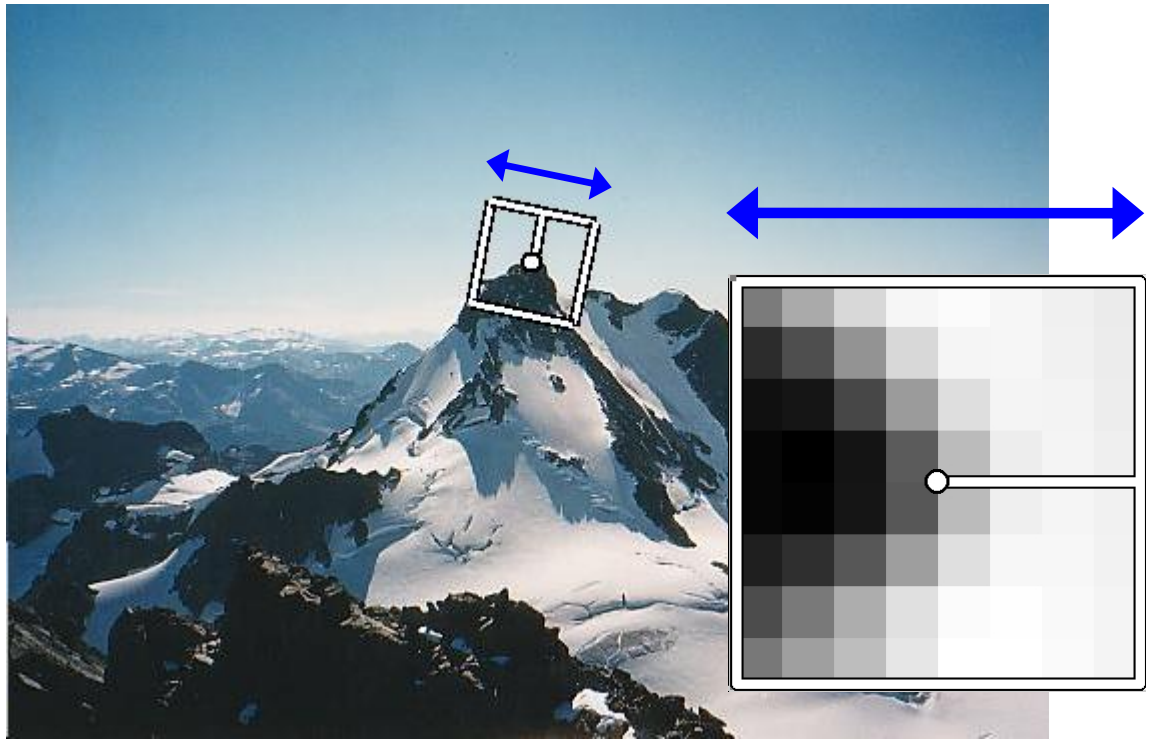
Invariance to rotation



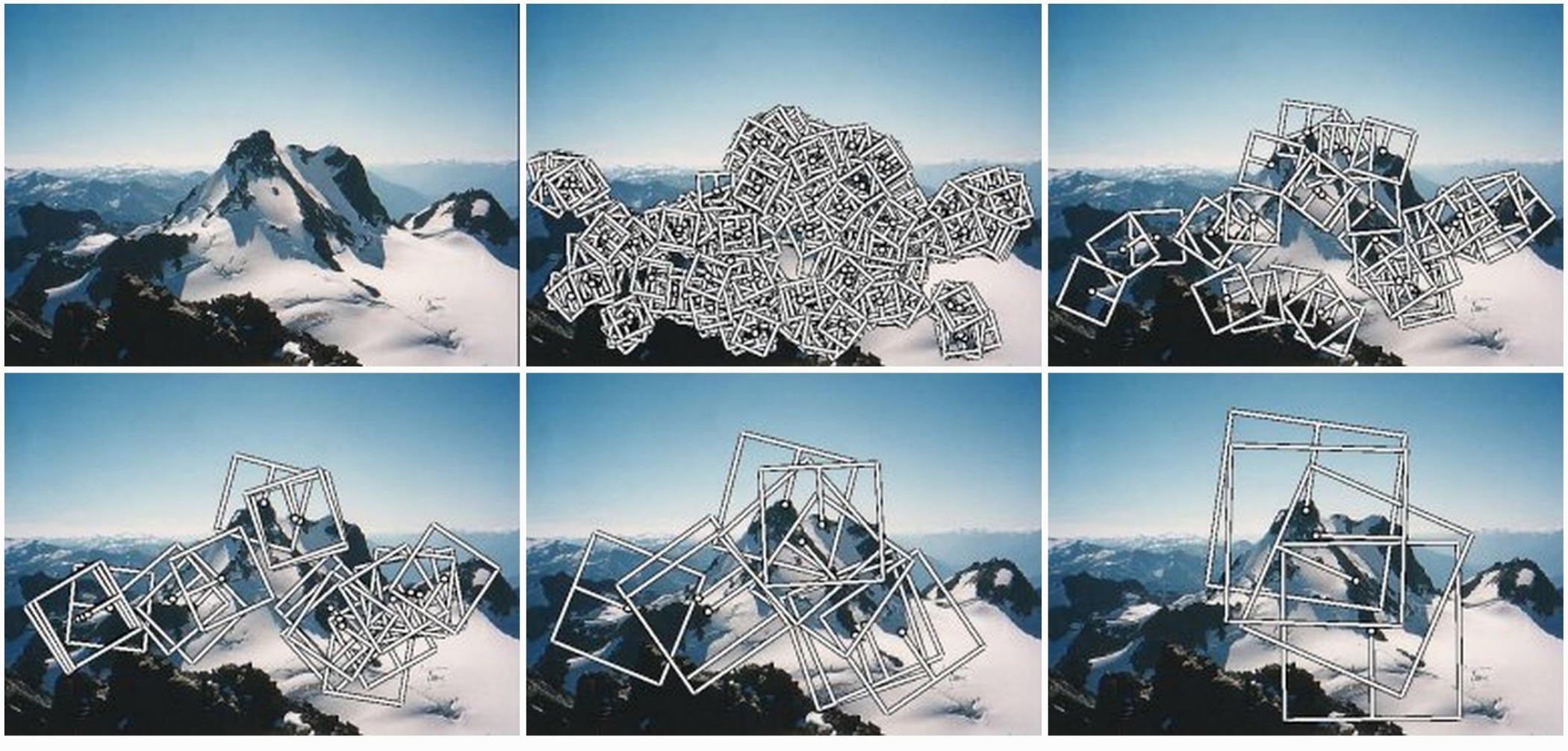


Invariance to rotation

- Rotate patch according to its dominant gradient orientation
- This puts the patches into a canonical orientation

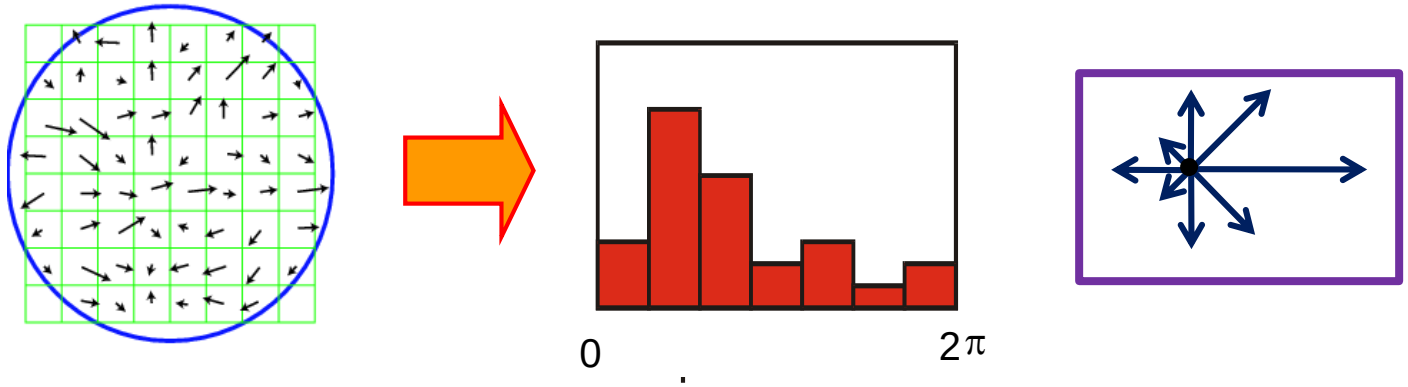


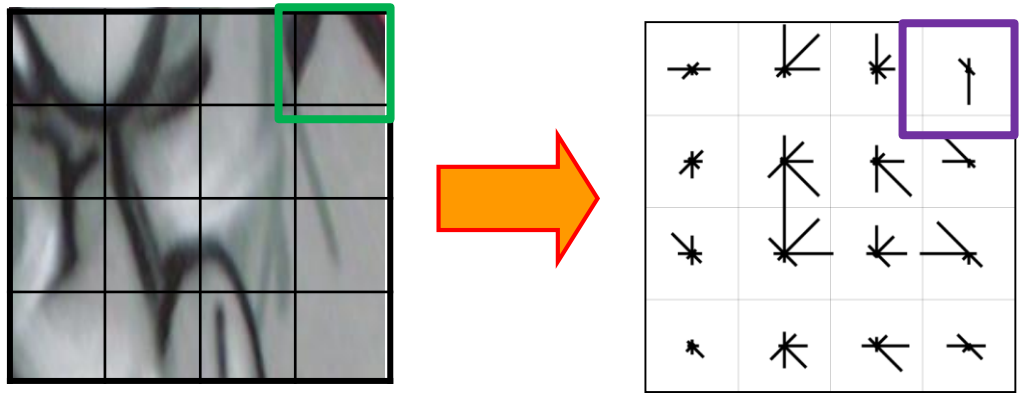
Multi-scale oriented patches (MOPS)



The SIFT descriptor

Use histograms to bin pixels within sub-patches according to their orientation.





Why sub-patches?

Why does SIFT have some illumination invariance?

Slide by James Hays

The SIFT descriptor

- Extraordinarily robust matching technique
- Can handle changes in viewpoint:
  - Up to ~60 degrees out-of-plane rotation
- Can handle significant illumination changes
  - Sometimes even day vs. night (below)
- Fast and efficient – can run in real time
- Implementation is part of your coursework

Slide by Steve Seitz

Speeded Up Robust Features (SURF)

For computational efficiency only compute gradient histogram with 4 bins:

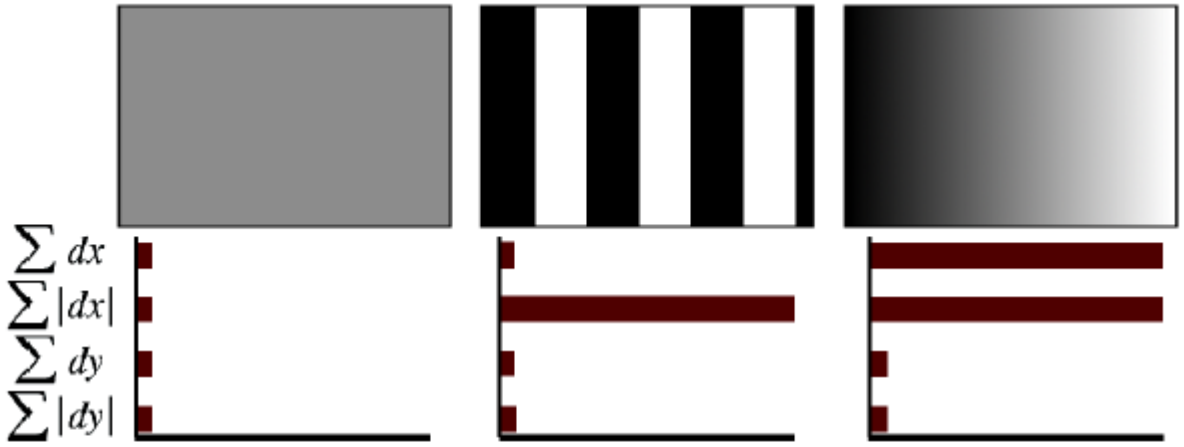


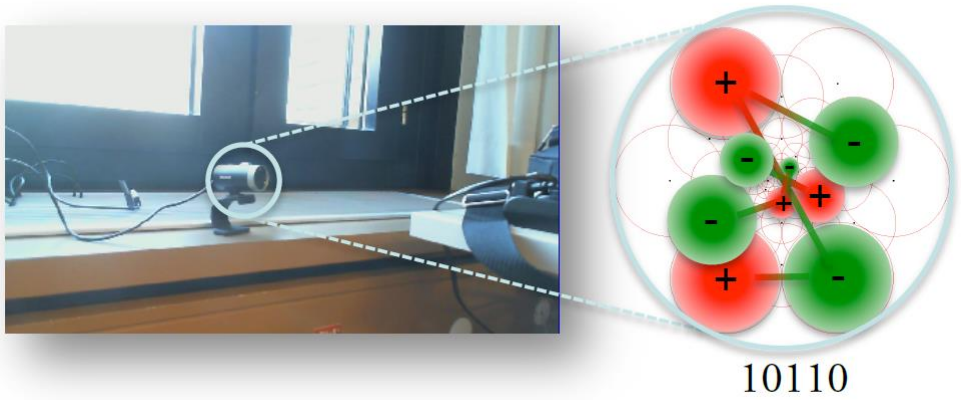
Fig. 3. The descriptor entries of a sub-region represent the nature of the underlying intensity pattern. Left: In case of a homogeneous region, all values are relatively low. Middle: In presence of frequencies in x direction, the value of $\sum |d_x|$ is high, but all others remain low. If the intensity is gradually increasing in x direction, both values $\sum d_x$ and $\sum |d_x|$ are high.

H. Bay, A. Ess, T. Tuytelaars and L. Van Gool
SURF: Speeded Up Robust Features
CVIU, 2008, 110(3), 346–359, 2008

Slide by Larry Zitnick

More descriptors

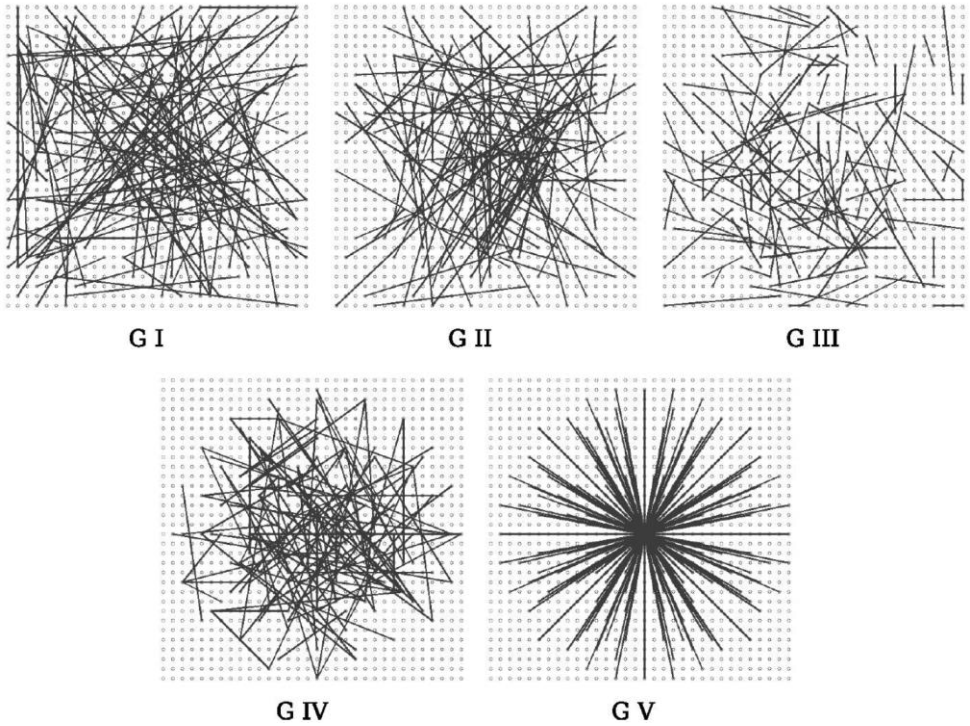
FREAK: Fast Retina Keypoint



Alahi et al. (CVPR 2012)

A. Alahi, R. Ortiz and P. Vandergheynst
FREAK: Fast Retina Keypoint
CVPR 2012

BRIEF: local binary descriptor



Calonder et al. (PAMI 2012)

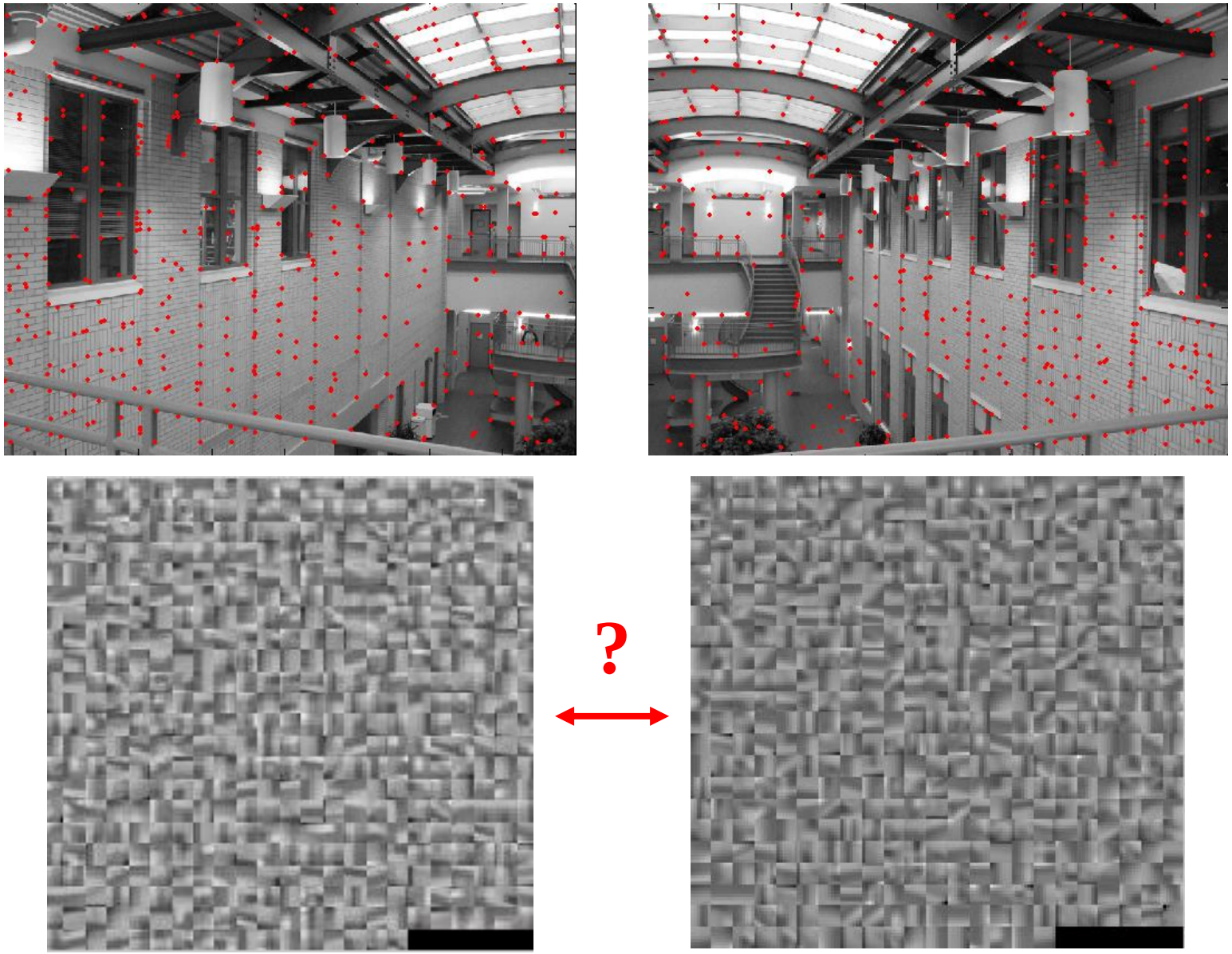
M. Calonder, V. Lepetit, M. Özuysal,
T. Trzcinski, C Strecha and P. Fua
BRIEF: Computing a Local Binary Descriptor Very Fast
IEEE TPAMI, 2012, 34(7), 1281–1298

Feature matching

Descriptor distance

- L1 (sum of absolute differences) / L2 (Euclidean distance):
  - SIFT: Scale-Invariant Feature Transform (Lowe, IJCV 2004)
  - SURF: Speeded-Up Robust Features (Bay et al., ECCV 2006, CVIU 2008)
- Hamming distance (for binary descriptors):
  - BRIEF: Binary Robust Independent Elementary Features (Calonder et al., ECCV 2010, PAMI 2012)
  - ORB: oriented BRIEF (Rublee et al., ICCV 2011)
  - BRISK: Binary Robust Invariant Scalable Keypoints (Leutenegger et al., ICCV 2011)
  - FREAK: Fast Retina Keypoint (Alahi et al., CVPR 2012)

Feature matching



Slide by Alexei Efros, Rick Szeliski and/or Steve Seitz

Feature matching

- Brute-force matching:
  - For each descriptor in the first set, find the closest descriptor in the second set **by trying each one**
  - Given 1000 features in each of two images, need to test one million pairs of features potential matches
- locality-sensitive hashing (LSH)
- randomised kd-trees
- k-means clustering tree

The FLANN library

- FLANN = "Fast Library for Approximate Nearest Neighbours"
- Implements different indices and algorithms, e.g. brute force, randomised kd-trees, k-means clustering tree
- **autotuned:** automatically determines the best index and optimum parameters using a cross-validation technique
- Website: http://www.cs.ubc.ca/research/flann/
- Also incorporated in OpenCV: `FlannBasedMatcher`

Muja M. & Lowe, D. G.
Scalable Nearest Neighbor Algorithms for High Dimensional Data
IEEE TPAMI, 2014, 36(11), 2227–2240

Cross checking matched features

- Only keep a match (a, b) if:
  - b is the best match for a
  - a is the best match for b
- Returns only consistent pairs
- Minimal number of outliers if there are enough initial matches

Ratio test (Lowe, 2004)

- Consider two best-matched features (m_1, m_2) to a feature f
- Ratio r = d(f, m_1) / d(f, m_2)
- If r is high, then f is
  - not unique (>1 match), or
  - not matched (random match)
- Rejects matches with r > 0.8
  - Eliminates 90% of false matches
  - Discards <5% of correct matches

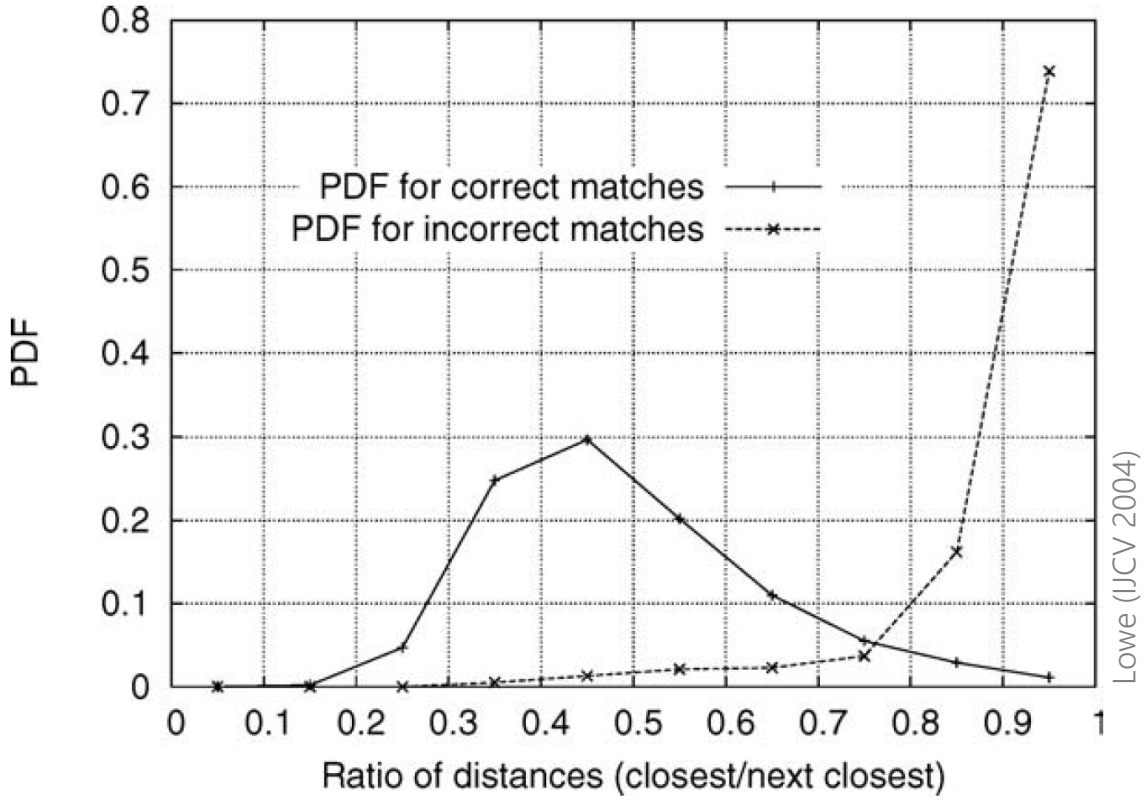
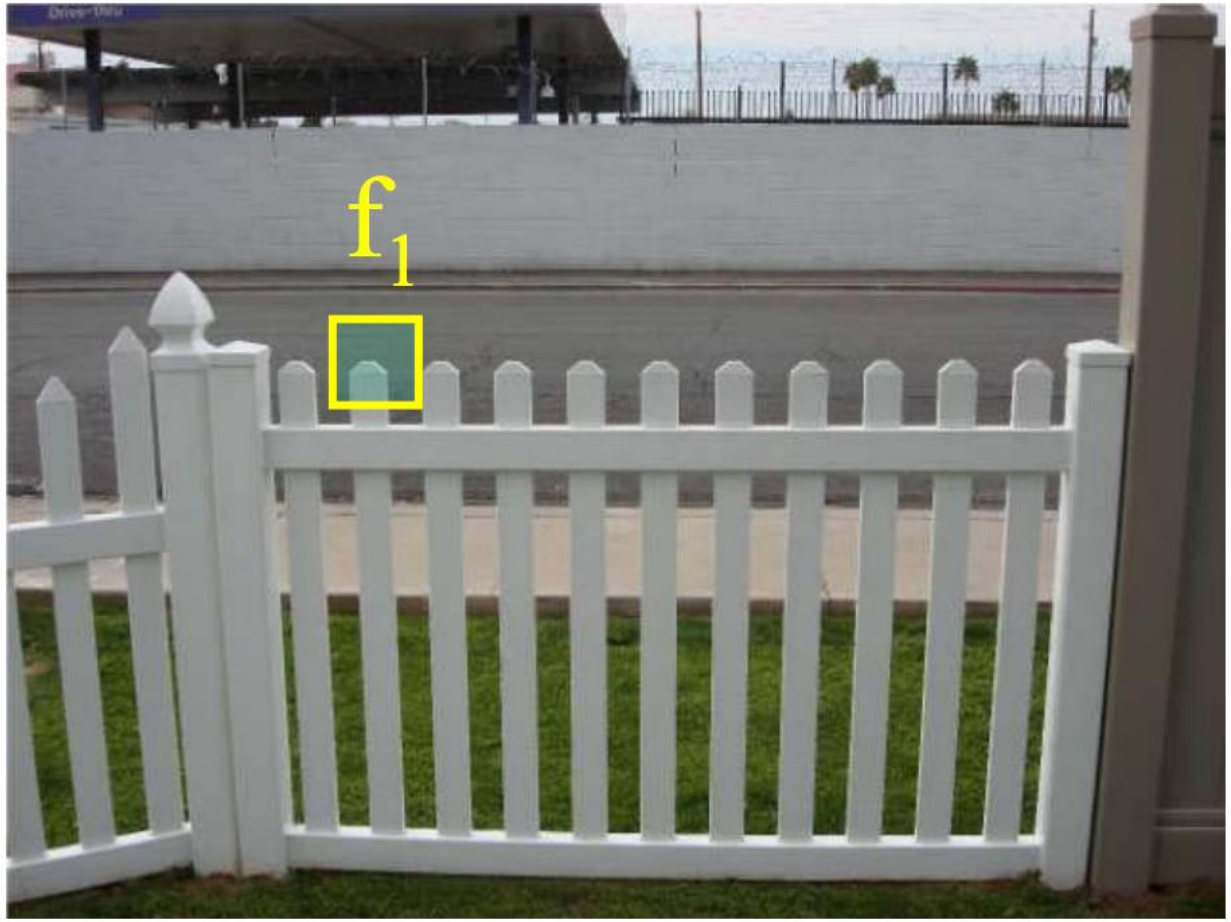
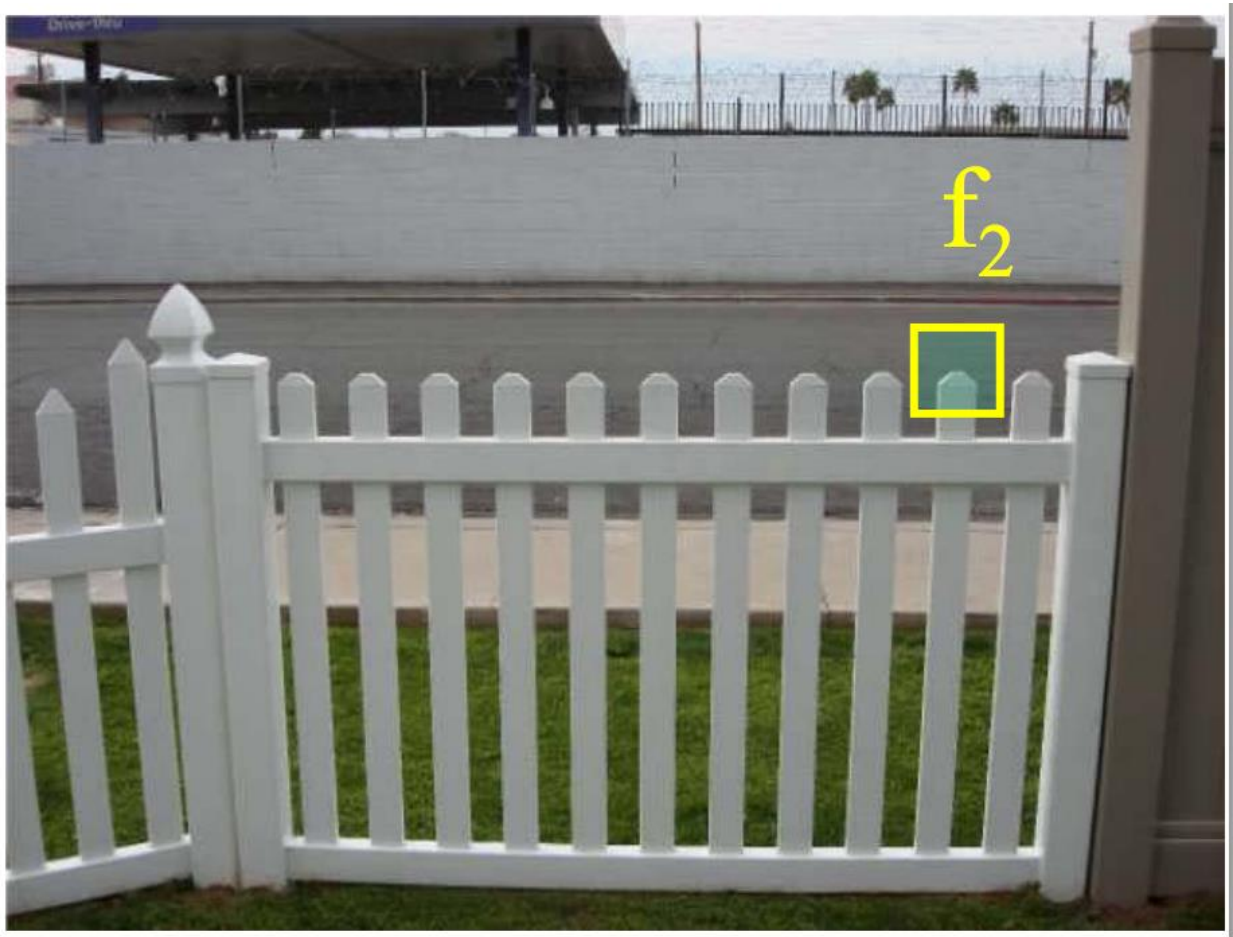


Figure 11. The probability that a match is correct can be determined by taking the ratio of distance from the closest neighbor to the distance of the second closest. Using a database of 40,000 keypoints, the solid line shows the PDF of this ratio for correct matches, while the dotted line is for matches that were incorrect.

Ratio test (Lowe, 2004)





Questions?