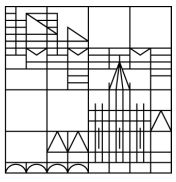
Chapter 4

Scale Invariant Feature Transform (SIFT)



University of
Konstanz

Lecture “Image Analysis and Computer Vision”
Winter semester 2014/15

Bastian Goldlücke

Overview

Overview

This chaper: (the first two-thirds of) a single paper

David Lowe, "Distinctive Image Features from Scale-Invariant Keypoints"

International Journal of Computer Vision 60(2), 91-110, 2004.

Note: I have included some additional explanations where appropriate.

Looks like a lot of citations ...

David Lowe

Follow

Professor of Computer Science, University of British Columbia
Computer Vision, Object Recognition
Verified email at cs.ubc.ca - Homepage

Title 1–20	Cited by	Year
Distinctive image features from scale-invariant keypoints DG Lowe International journal of computer vision 60 (2), 91-110	26512	2004
Object recognition from local scale-invariant features DG Lowe International Conference on Computer Vision, 1999, 1150-1157	8349	1999
Perceptual Organization and Visual Recognition. DG Lowe STANFORD UNIV CA DEPT OF COMPUTER SCIENCE	1529	1984
Three-dimensional object recognition from single two-dimensional images DG Lowe Artificial intelligence 31 (3), 355-395	1444	1987
Fitting parameterized three-dimensional models to images DG Lowe IEEE Transactions on Pattern Analysis and Machine Intelligence 13 (5), 441-450	1010	1991
Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration. M Muja, DG Lowe VISAPP (1), 331-340	963	2009
Recognising panoramas M Brown, DG Lowe International Conference on Computer Vision, 2003, 1218-1225	927	2003

Google Scholar

Citation indices	All	Since 2009
Citations	51724	36488
h-index	45	36
i10-index	81	54

Co-authors View all…

James Little
Marius Muja
Matthew Brown
David Meger
Sancho McCann
Per-Erik Forssén
Peter D. Lawrence
Nando de Freitas
Kevin Lai
Dinesh K. Pai
Gustavo Carneiro
Michael C Yip
S.E. Salcudean
Robert Rohling

... to put that into perspective ...

Albert Einstein

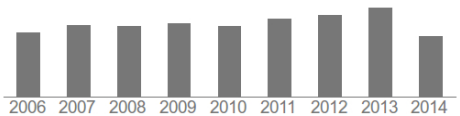
Institute of Advanced Studies, Princeton
Physics
No verified email

Follow

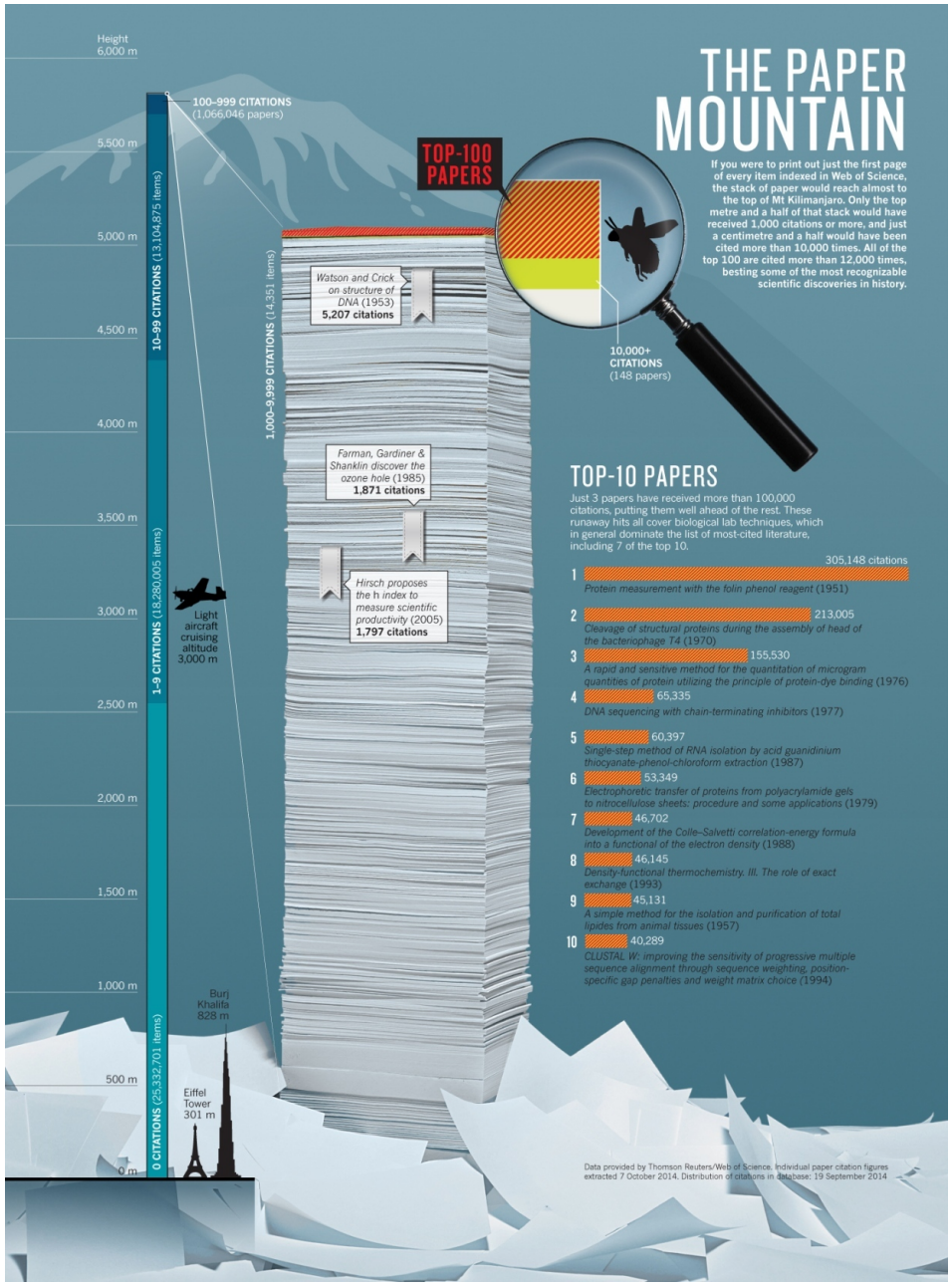
Google Scholar

Title 1–20	Cited by	Year
Can quantum-mechanical description of physical reality be considered complete? A Einstein, B Podolsky, N Rosen Physical review 47 (10), 777	12711	1935
Uber einen die Erzeugung und Verwandlung des Lichtes betreffenden heurischen Gesichtpunkt A Einstein Ann. Phys. 17, 132-148	7070 *	1905
On the movement of small particles suspended in stationary liquids required by the molecular-kinetic theory of heat A Einstein Annalen der Physik 17, 549-560	5618 *	1905
Zur Elektrodynamik bewegter Körper A Einstein	3756 *	
Investigations on the Theory of the Brownian Movement A Einstein Dover publications	3337	1956
Eine neue bestimmung der moleküldimensionen A Einstein Annalen der Physik 324 (2), 289-306	2956	1906
The meaning of relativity		

Citation indices	All	Since 2009
Citations	86199	28028
h-index	103	62
i10-index	361	197



.. a cool visualization of citation counts over all papers



http://www.nature.com/news/nature-top-100-papers-infographicv2-30-10-14-jpg-7.21204?article=1.16224

What's in it that seems so useful?

Distinctive Image Features from Scale-Invariant Keypoints

Goals

- detect characteristic feature points in images (mainly corners)
- create a detailed descriptor that represents each feature as unique as possible
- ensure invariance with respect to changes in location, scale and orientation

Applications

- particularly useful for finding sparse correspondences between images
- mainly used in computer vision:
  - estimation of global relations between images (stereo geometry, homographies, parametric transformations)
  - tracking and structure-from-motion
  - object detection and recognition

The Scale-Invariant Feature Transform (SIFT)

- **Step 1: detection of characteristic feature points**
  - based on a Gaussian scale-space using difference of Gaussians
  - extrema provide location and scale

The Scale-Invariant Feature Transform (SIFT)

- **Step 1: detection of characteristic feature points**
  - based on a Gaussian scale-space using difference of Gaussians
  - extrema provide location and scale
- **Step 2: accurate localisation of key points**
  - performs sub-pixel refinement by fitting quadratic functions
  - additionally discards points with high ratio between principal curvatures

The Scale-Invariant Feature Transform (SIFT)

- **Step 1: detection of characteristic feature points**
  - based on a Gaussian scale-space using difference of Gaussians
  - extrema provide location and scale
- **Step 2: accurate localisation of key points**
  - performs sub-pixel refinement by fitting quadratic functions
  - additionally discards points with high ratio between principal curvatures
- **Step 3: assignment of the dominant orientation(s)**
  - based on a histogram of gradients within the local neighbourhood
  - refines orientation by fitting quadratic functions

The Scale-Invariant Feature Transform (SIFT)

- **Step 1: detection of characteristic feature points**
  - based on a Gaussian scale-space using difference of Gaussians
  - extrema provide location and scale
- **Step 2: accurate localisation of key points**
  - performs sub-pixel refinement by fitting quadratic functions
  - additionally discards points with high ratio between principal curvatures
- **Step 3: assignment of the dominant orientation(s)**
  - based on a histogram of gradients within the local neighbourhood
  - refines orientation by fitting quadratic functions
- **Step 4: computation of a suitable key point descriptor**
  - unit vector based on accumulated histograms of gradients
  - compensated by location, scale and dominant orientation

- None of the steps is overly difficult, and you have alreadly learned enough about image processing to understand everything about SIFT feature extraction and matching.
- However, a lot of engineering is involved to get everything working at maximum possible quality, which is maybe the major contribution of the paper.

Overview

Step 1: detection of characteristic feature points

- Note: this is complementary to the techniques we already know (e.g. Harris/Foerstner detector), which can be used just as well.
- However, Harris/Foerstner needs to be properly extended to scale space to make responses at different scales comparable, which we do not pursue in detail here.

Detection of scale space extrema

Starting point: Gaussian Scale-Space of input image f

- discrete levels of smoothing $\sigma_0, \sigma_1, \ldots, \sigma_t, \ldots, \sigma_{\max}$
- consecutive levels related by a constant factor, $\sigma_{t+1} = k\sigma_t$, $k > 1$.
- scale-space given by convolution of f with increasing Gaussians G_{σ_t}:

$$f_t = G_{\sigma_t} * f.$$

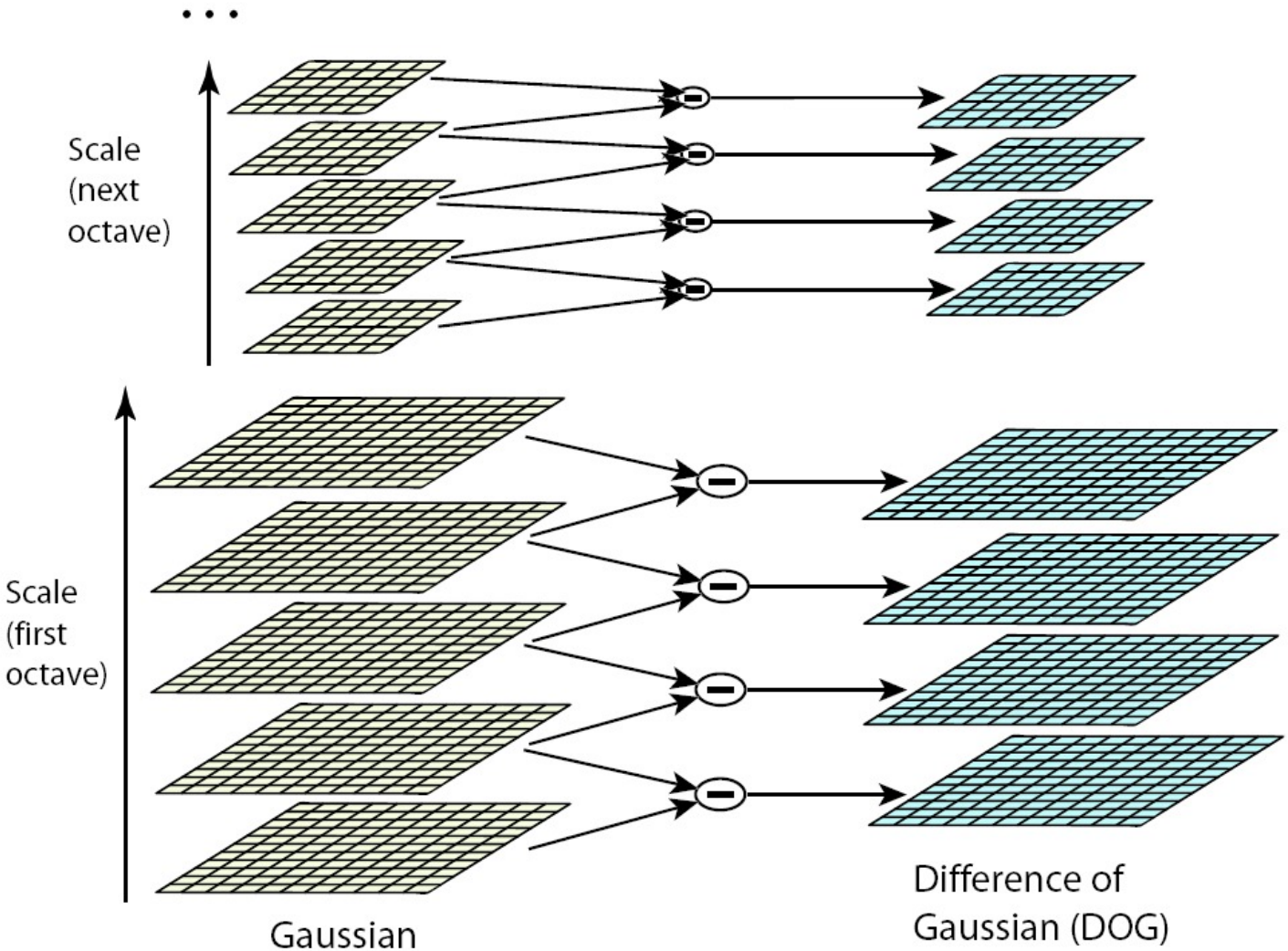
From this: difference-of-Gaussians (DoG)

- difference of two consecutive scales of the Gaussian scale-space

$$\begin{aligned} D_t :&= f_{t+1} - f_t \\ &= G_{k\sigma_t} * f - G_{\sigma_t} * f \\ &= (G_{k\sigma_t} - G_{\sigma_t}) * f. \end{aligned}$$

- Reminder: the difference of two Gaussians is a bandpass filter, only details of a certain scale or frequency range survive.
- Idea: feature generates the strongest detector response at a certain scale, search for maxima across scales to achieve scale invariance.

SIFT scale space pyramid



Computation of the difference of Gaussians from Gaussian scale-space for SIFT feature detection. In this example, the resolution is halved every octave for performance reasons. The original implementation uses five scales per octave, $\sigma_0 = 0.5$ and $\rho = \sqrt{2}$.

DoG and the Laplacian-of-Gaussian filter

- The DoG is related to the scale derivative of the Gaussian via the following finite difference approximation:

$$\partial_\sigma G_\sigma \approx \frac{G_{k\sigma} - G_\sigma}{k\sigma - \sigma}$$

DoG and the Laplacian-of-Gaussian filter

- The DoG is related to the scale derivative of the Gaussian via the following finite difference approximation:

$$\partial_\sigma G_\sigma \approx \frac{G_{k\sigma} - G_\sigma}{k\sigma - \sigma}$$

- Moreover, the analytic scale derivative of the Gaussian is the scaled **Laplacian-of-Gaussian (LoG)**:

$$\partial_\sigma G_\sigma = \sigma \Delta G_\sigma = \sigma(\partial_x^2 G_\sigma + \partial_y^2 G_\sigma).$$

“Proof by Mathematica”: scale derivative of Gaussian

WolframAlpha computational knowledge engine

simplify d/ds 1 / (2 pi s^2) exp(- (x^2 + y^2) / (2 s^2))

Examples Random

Input interpretation:

simplify $\frac{\partial}{\partial s}\left(\frac{1}{2\pi s^2}\exp\left(-\frac{x^2+y^2}{2s^2}\right)\right)$

Results:

$$-\frac{e^{-\frac{x^2}{2}-\frac{y^2}{2}}}{s^2}\left(2s^2-x^2-y^2\right)$$

$$\frac{x^2 e^{-\frac{x^2+y^2}{2s^2}}}{2\pi s^5}+\frac{y^2 e^{-\frac{x^2+y^2}{2s^2}}}{2\pi s^5}-\frac{e^{-\frac{x^2+y^2}{2s^2}}}{\pi s^3}$$

“Proof by Mathematica”: Laplacian of Gaussian part 1

WolframAlpha computational knowledge engine

simplify d/dx d/dx 1 / (2 pi s^2) exp(- (x^2 + y^2) / (2 s^2))

Examples Random

Input interpretation:

simplify $\frac{\partial}{\partial x}\frac{\partial}{\partial x}\left(\frac{1}{2\pi s^2}\exp\left(-\frac{x^2+y^2}{2s^2}\right)\right)$

Results: More

$$-\frac{(s-x)(s+x)e^{-\frac{x^2}{2s^2}-\frac{y^2}{2s^2}}}{2\pi s^6}$$

$$-\frac{\left(s^2-x^2\right)e^{\frac{-\frac{x^2}{2}-\frac{y^2}{2}}{s^2}}}{2\pi s^6}$$

$$\frac{x^2 e^{-\frac{x^2+y^2}{2s^2}}}{2\pi s^6}-\frac{e^{-\frac{x^2+y^2}{2s^2}}}{2\pi s^4}$$

“Proof by Mathematica”: Laplacian of Gaussian part 2

WolframAlpha computational knowledge engine

simplify d/dy d/dy 1 / (2 pi s^2) exp(- (x^2 + y^2) / (2 s^2))

Examples Random

Input interpretation:

simplify $\frac{\partial}{\partial y}\frac{\partial}{\partial y}\left(\frac{1}{2\pi s^2}\exp\left(-\frac{x^2+y^2}{2s^2}\right)\right)$

Results: More

$$-\frac{(s-y)(s+y)\,e^{-\frac{x^2}{2s^2}-\frac{y^2}{2s^2}}}{2\pi s^6}$$

$$-\frac{\left(s^2-y^2\right)e^{\frac{-\frac{x^2}{2}-\frac{y^2}{2}}{s^2}}}{2\pi s^6}$$

$$\frac{y^2\,e^{-\frac{x^2+y^2}{2s^2}}}{2\pi s^6}-\frac{e^{-\frac{x^2+y^2}{2s^2}}}{2\pi s^4}$$

Relation between DoG and LoG

From the previous two formulas

$$\partial_\sigma G_\sigma \approx \frac{G_{k\sigma} - G_\sigma}{k\sigma - \sigma} \qquad \text{and} \qquad \partial_\sigma G_\sigma = \sigma \Delta G_\sigma = \sigma(\partial_x^2 G_\sigma + \partial_y^2 G_\sigma),$$

we see that in summary, the DoG is an approximation to the scaled LoG up to a constant factor:

$$\boxed{G_{k\sigma} - G_\sigma \approx (k-1)\sigma^2 \Delta G_\sigma}$$

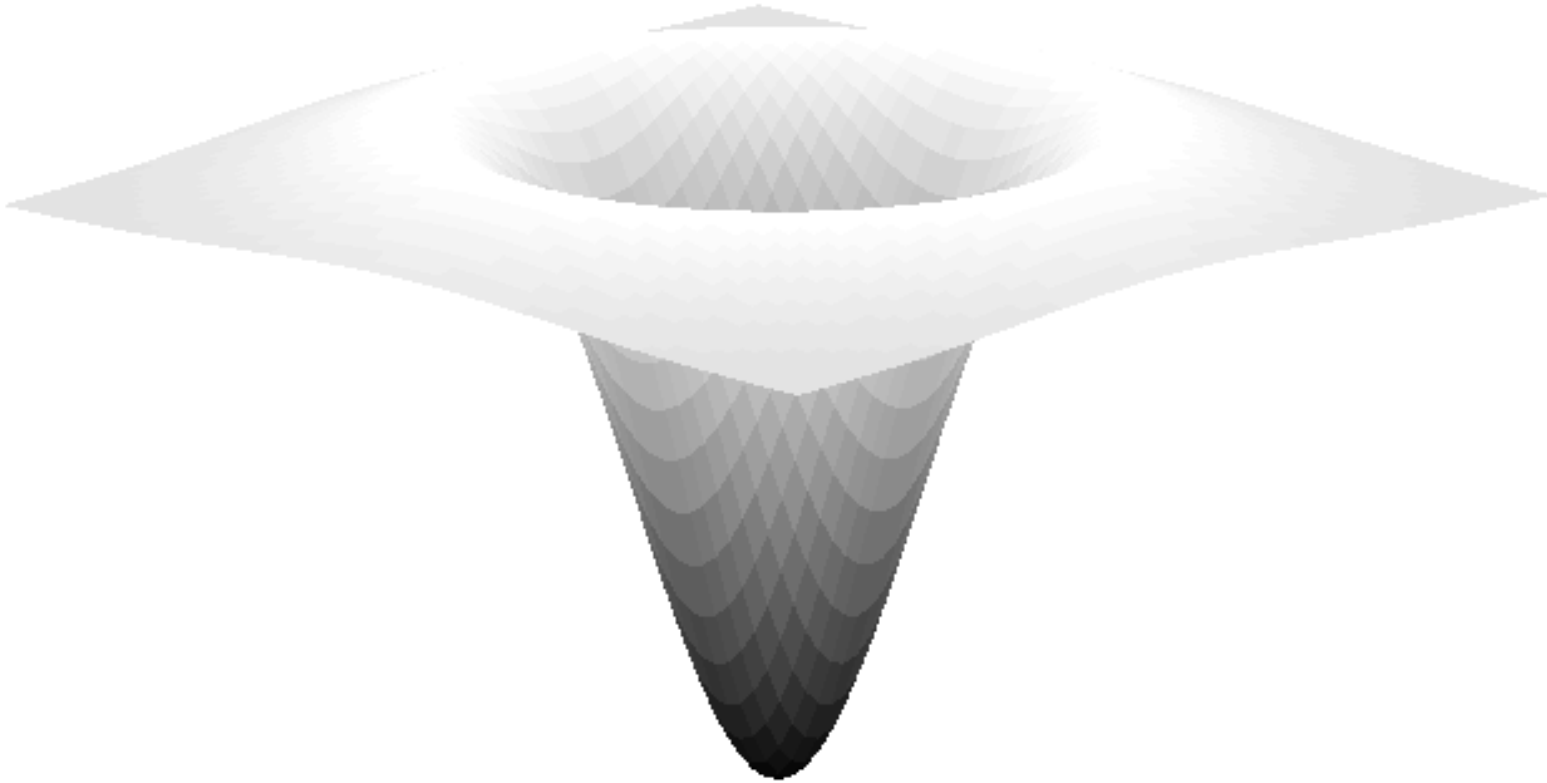
The factor $(k-1)$ can be neglected, since it is independent of the scale σ and thus does not influence the location of extremal points in scale-space.

The operator $\sigma^2 \Delta G_\sigma$ is called **normalised Laplacian-of-Gaussian.**

Side notes:

- *this shows equivalence of scale space construction to heat diffusion.*
- *it also explains where the Laplacian pyramid has its name from.*

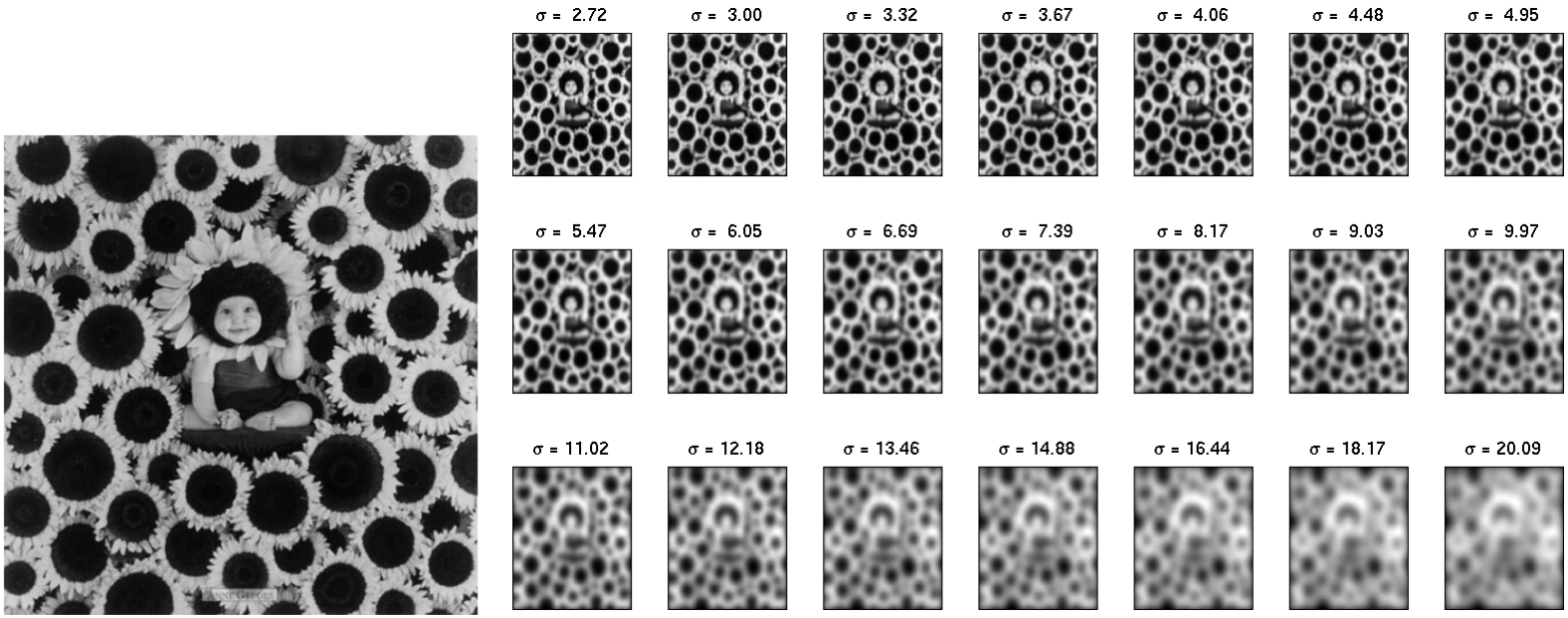
The normalized Laplacian of Gaussian filter



Remember that filters detect the shape they look like?
The LoG detects “**blobs**” !

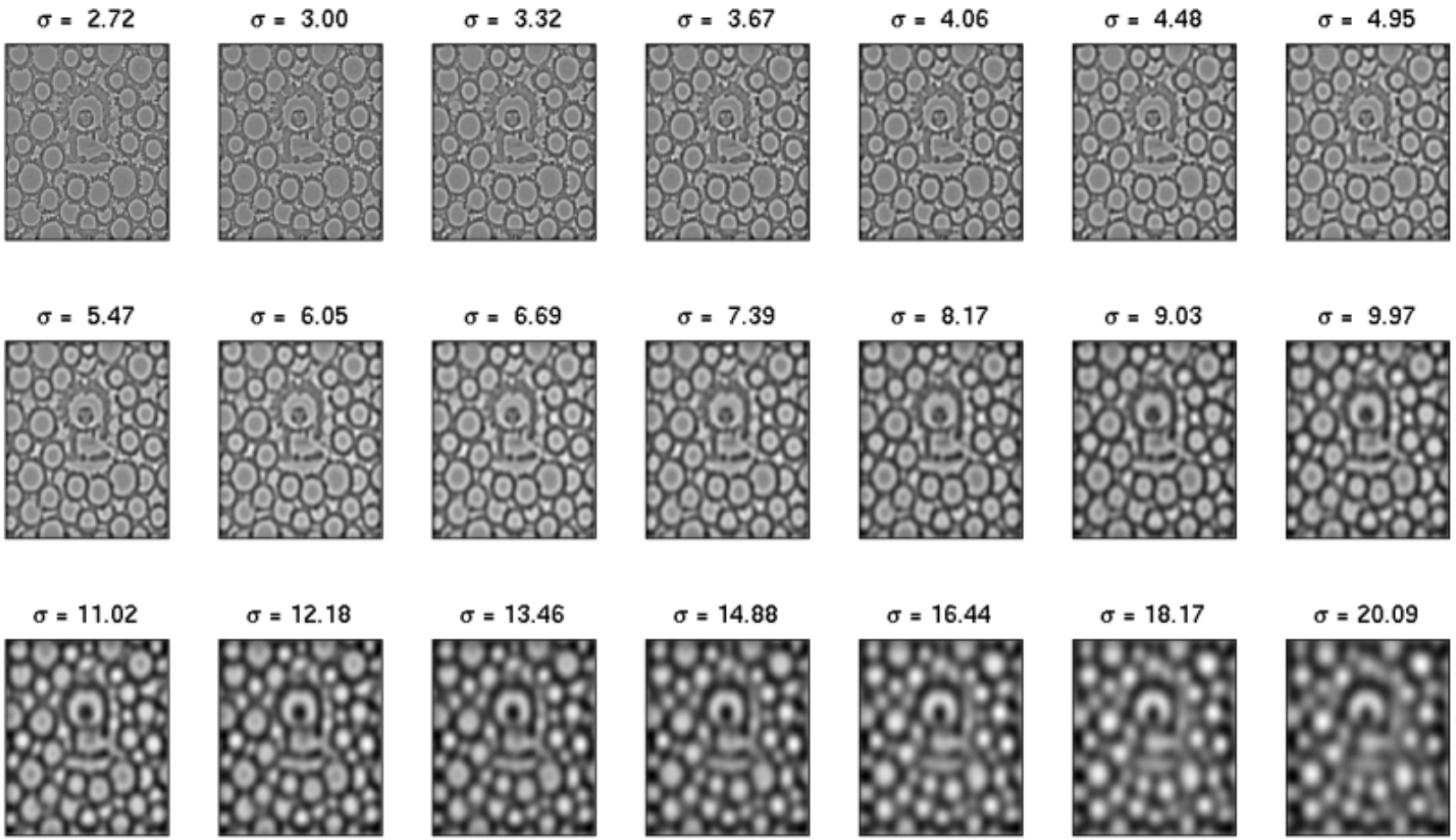
Images: J. Fishbaugh, http://www.cs.utah.edu/~jfishbau/advimproc/project1/

Example: image and scale space



Images: J. Fishbaugh, http://www.cs.utah.edu/~jfishbau/advimproc/project1/

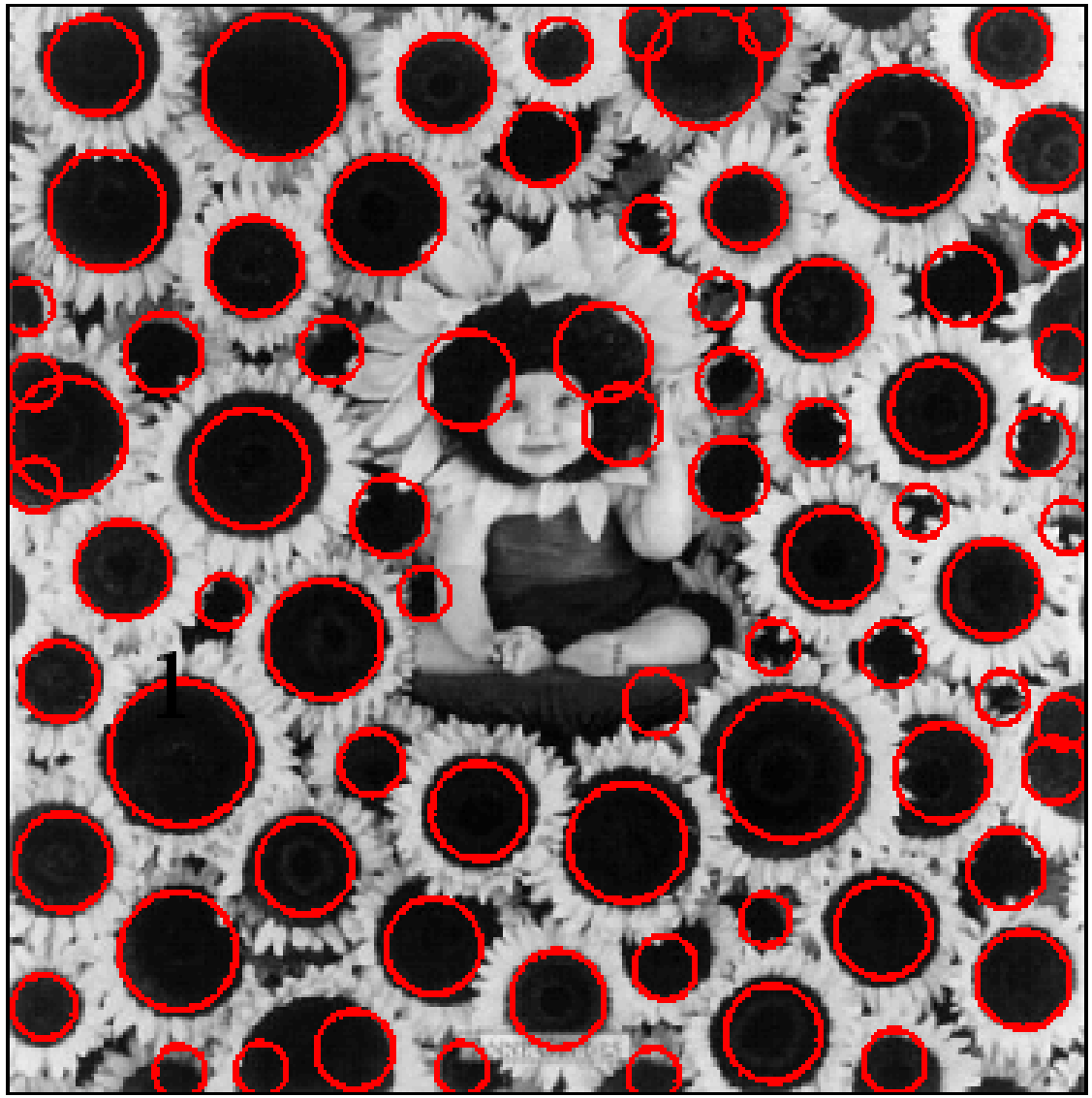
Example: normalized Laplacian of Gaussian



Strong responses for blobs,
scale of maximum response depends on blob size

Images: J. Fishbaugh, http://www.cs.utah.edu/~jfishbau/advimproc/project1/

Example: blob detection results



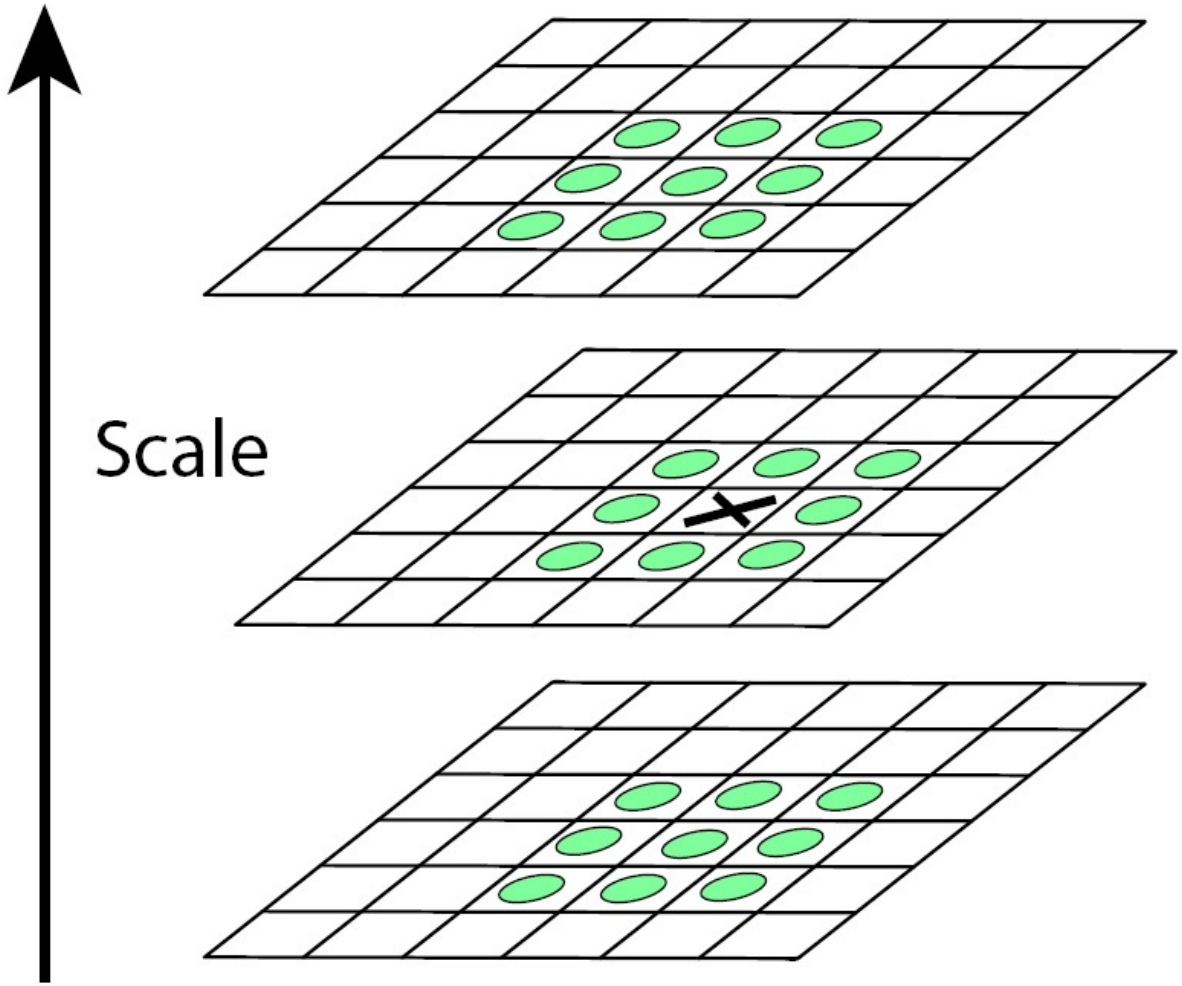
Detected blobs after non-maxima suppression (spatial/scale), circles with radius equal to 1.5σ.

Images: J. Fishbaugh, http://www.cs.utah.edu/~jfishbau/advimproc/project1/

Feature detection based on the normalised LoG

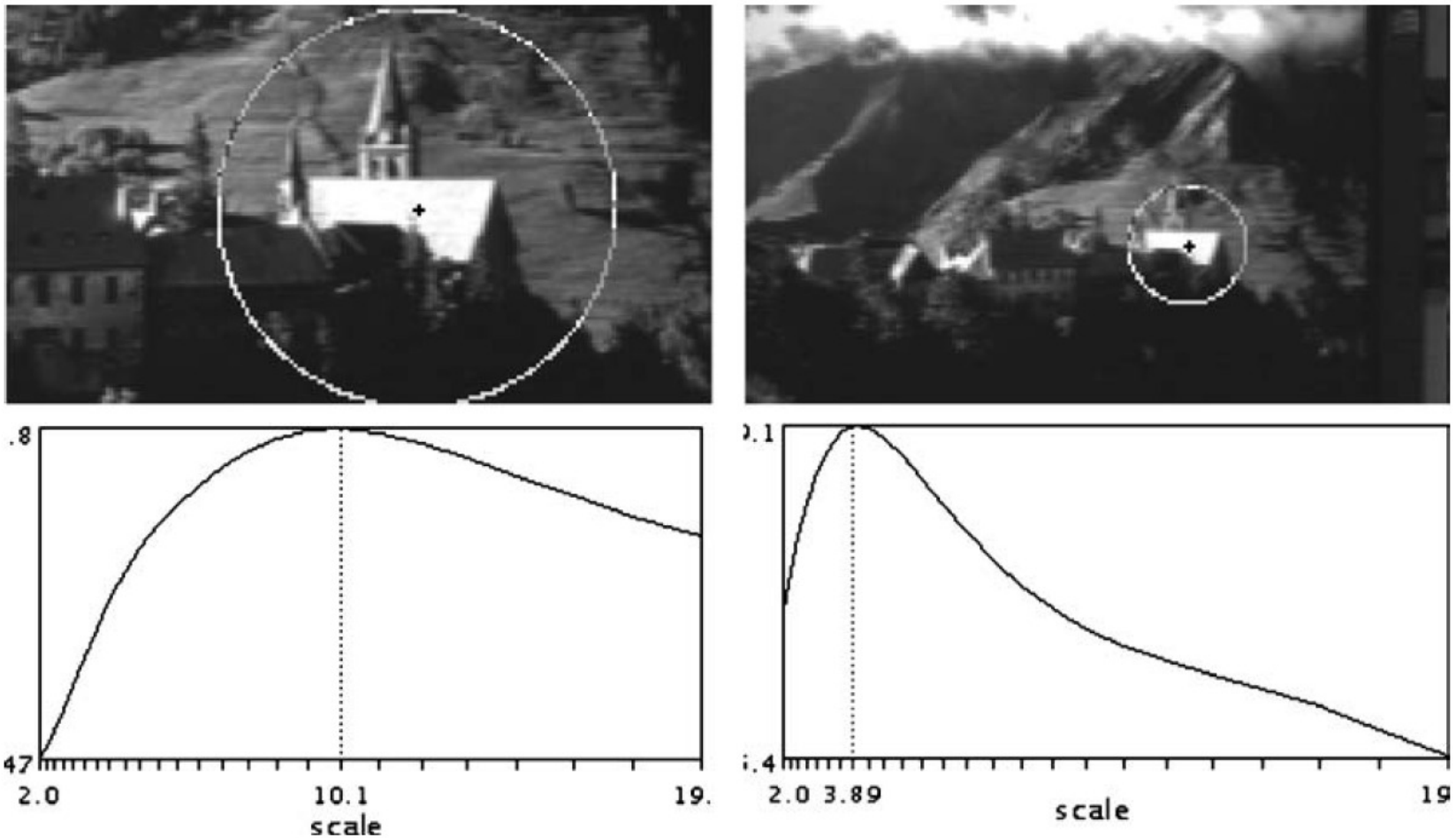
- Lindeberg (1994) proved that the factor σ^2 is required to achieve true scale invariance. It makes values from different scales of the LoG comparable.
- Since structures only live at a certain scale, they first appear in the LoG when coarsening the scale and then vanish again (bandpass property of the DoG).
- Thus, at the **characteristic scale** of a feature, the magnitude of the normalised LoG takes an extremal value (maximum or minimum!)
- By computing extrema jointly in scale and space, this detection steps provides both the **scale and the location** of image features.
- In the simplest case, extremal values of the normalised LoG can be found by comparing each value with its direct neighbours in scale and space.

Detection of scale-space extrema



Computation of extremal values by comparing the central value to all 26 local neighbours in the normalised LoG scale space.

Example: scale-space extrema under change of magnification



Top row: Images taken with two different focal lengths. **Bottom row:** Corresponding values of the magnitude of the normalised LoG over the different scales. The scale selected by the maxima of both graphs is depicted in both images as a circle with a radius that corresponds to three times the standard deviation (contains 99.7% of the information).

Step 2: accurate localisation of key points

Step 2a: sub-pixel refinement

- determines sub-pixel location of extremal values by performing a second-order Taylor expansion (polynomial fit) around each feature point $\mathbf{x}_i = (x_i, y_i, \sigma_i)$. For distance $\mathbf{h}$ to the feature,

$$D(\mathbf{x}_i + \mathbf{h}) \approx D(\mathbf{x}_i) + \nabla D(\mathbf{x}_i)^T \mathbf{h} + \mathbf{h}^T H(x_i) \mathbf{h}.$$

- The gradient ∇D and the Hessian $H = H(D)$ of the scale-space function can be approximated locally by central differences using neighbouring pixel values.

- Note: the Hessian is the symmetric 3×3 matrix of mixed second derivatives (actually a tensor, but no one admits that for some reason), the gradient a 3×1 column vector.

Step 2a: sub-pixel refinement

- The necessary condition for the extremum location is then given by setting the derivative of the quadratic approximation with respect to **h** to zero:

$$\nabla D(\mathbf{x}_i) + H(\mathbf{x}_i)\mathbf{h} = 0.$$

  You can try to derive that at home as a (masochistic) exercise.

Step 2a: sub-pixel refinement

- The necessary condition for the extremum location is then given by setting the derivative of the quadratic approximation with respect to **h** to zero:

$$\nabla D(\mathbf{x}_i) + H(\mathbf{x}_i)\mathbf{h} = 0.$$

  You can try to derive that at home as a (masochistic) exercise.

- Solving for **h** yields the offset to the extremum,

$$\hat{\mathbf{h}} = -H(\mathbf{x}_i)^{-1}\nabla D(\mathbf{x}_i).$$

Step 2a: sub-pixel refinement

- The necessary condition for the extremum location is then given by setting the derivative of the quadratic approximation with respect to **h** to zero:

$$\nabla D(\mathbf{x}_i) + H(\mathbf{x}_i)\mathbf{h} = 0.$$

  You can try to derive that at home as a (masochistic) exercise.

- Solving for **h** yields the offset to the extremum,

$$\hat{\mathbf{h}} = -H(\mathbf{x}_i)^{-1}\nabla D(\mathbf{x}_i).$$

- This allows to compute the new location and scale as $\hat{\mathbf{x}}_i = \mathbf{x}_i + \hat{\mathbf{h}}$.

Step 2a: sub-pixel refinement

- The necessary condition for the extremum location is then given by setting the derivative of the quadratic approximation with respect to **h** to zero:

$$\nabla D(\mathbf{x}_i) + H(\mathbf{x}_i)\mathbf{h} = 0.$$

  You can try to derive that at home as a (masochistic) exercise.

- Solving for **h** yields the offset to the extremum,

$$\hat{\mathbf{h}} = -H(\mathbf{x}_i)^{-1}\nabla D(\mathbf{x}_i).$$

- This allows to compute the new location and scale as $\hat{\mathbf{x}}_i = \mathbf{x}_i + \hat{\mathbf{h}}$.

- Finally, replacing **h** in the quadratic function with $\hat{\mathbf{h}}$ yields the new extremal value:

$$D(\hat{\mathbf{x}}_i) = D(\mathbf{x}_i) + \frac{1}{2}\nabla D(\mathbf{x}_i)^T\hat{\mathbf{h}}.$$

Step 2a: sub-pixel refinement

- The necessary condition for the extremum location is then given by setting the derivative of the quadratic approximation with respect to **h** to zero:

$$\nabla D(\mathbf{x}_i) + H(\mathbf{x}_i)\mathbf{h} = 0.$$

  You can try to derive that at home as a (masochistic) exercise.

- Solving for **h** yields the offset to the extremum,

$$\hat{\mathbf{h}} = -H(\mathbf{x}_i)^{-1}\nabla D(\mathbf{x}_i).$$

- This allows to compute the new location and scale as $\hat{\mathbf{x}}_i = \mathbf{x}_i + \hat{\mathbf{h}}$.

- Finally, replacing **h** in the quadratic function with $\hat{\mathbf{h}}$ yields the new extremal value:

$$D(\hat{\mathbf{x}}_i) = D(\mathbf{x}_i) + \frac{1}{2}\nabla D(\mathbf{x}_i)^T\hat{\mathbf{h}}.$$

- Using the above steps for each potential feature, the accuracy of feature locations is improved w.r.t. both scale and location.

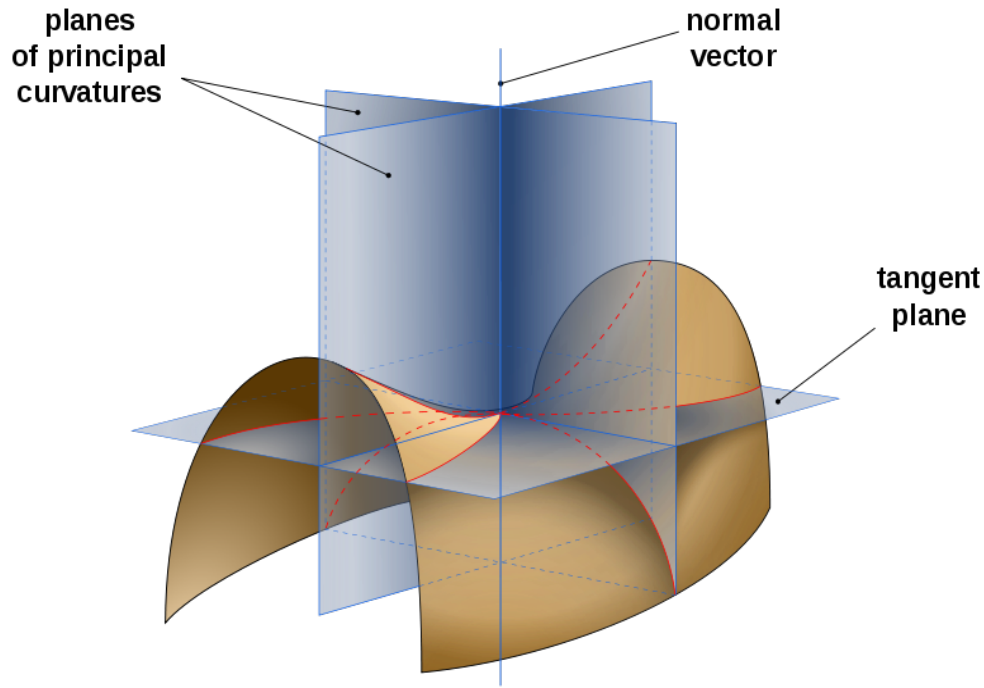
Step 2b: elimination of weak detections

- Feature point candidates are discarded if $D(\hat{\mathbf{x}}_i)$ is below a certain threshold (in the paper, it's set to around 0.03).

- This eliminates unreliable features in regions of low contrast.

Step 2c: elimination of unstable features

- Based on “principal curvatures” of the graph of the LoG image:



- Intuitively, the principal curvatures measure how the surface bends by different amounts in different directions.
- We want the smallest and largest amount of bending have the same sign and about the same magnitude, then the extremum is stable (i.e. well localized).

Image: Wikipedia “Principal curvature”

Step 2c: elimination of unstable features

- The eigenvalues $\lambda_1 \geq \lambda_2$ of the spatial Hessian

$$H_2 = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{yx} & D_{yy} \end{bmatrix}$$

  are proportional to the principal curvatures at the corresponding point.

- The following expression is minimised for small ratios $r = \lambda_1/\lambda_2 \geq 1$ and does not require to compute the eigenvalues explicitly:

$$\kappa = \frac{\mathrm{trace}(H_2)^2}{\det(H_2)} = \frac{(\lambda_1 + \lambda_2)^2}{\lambda_1 \lambda_2} = \frac{(r+1)^2}{r} = r + 2 + \frac{1}{r}.$$

- Feature point candidates are discarded if:
  - $\det(H_2) < 0$, correspond to saddle points instead of extrema (curvatures have opposite sign).
  - $\det(H_2) = 0$, no extremum either.
  - $\kappa > 10$, looks like ridge or valley instead of well-localized peak.

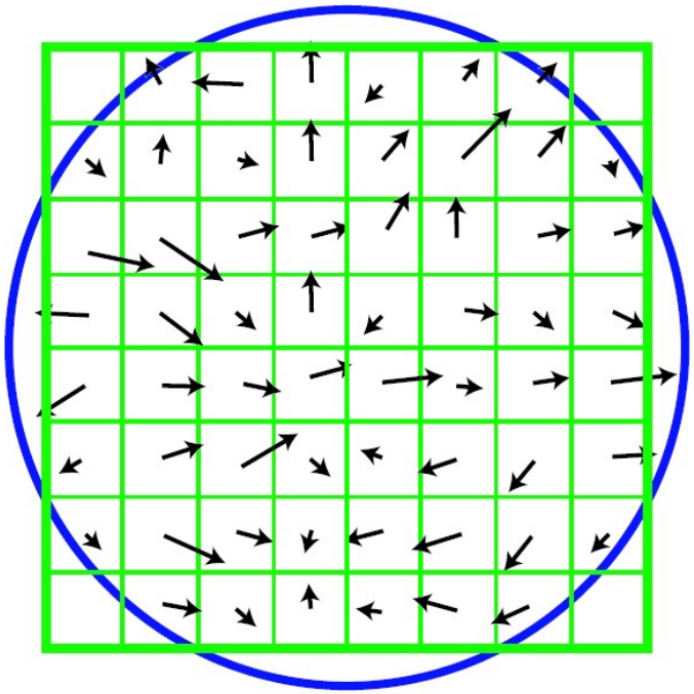
you have now learned some differential geometry - no area of mathematics will remain untouched at the end of the semester ;).

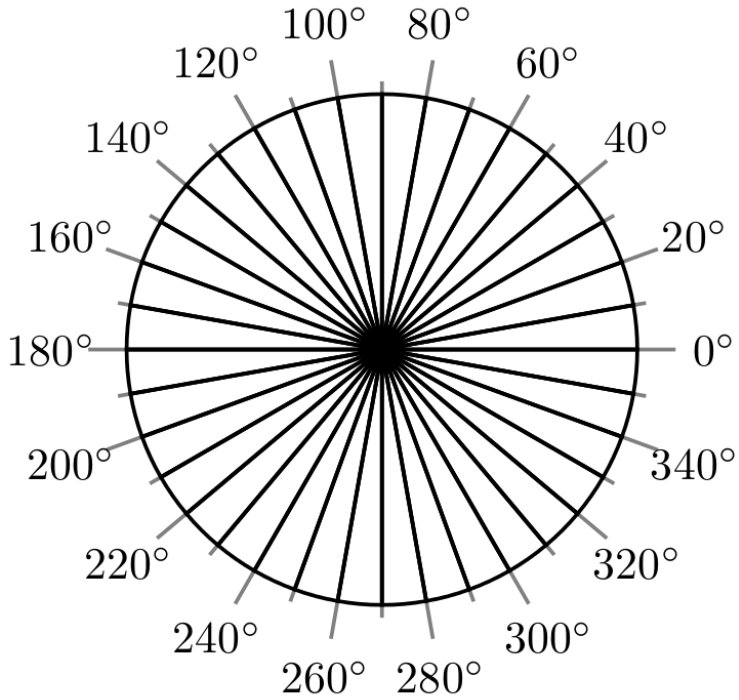
Step 3: assignment of the dominant orientation(s)

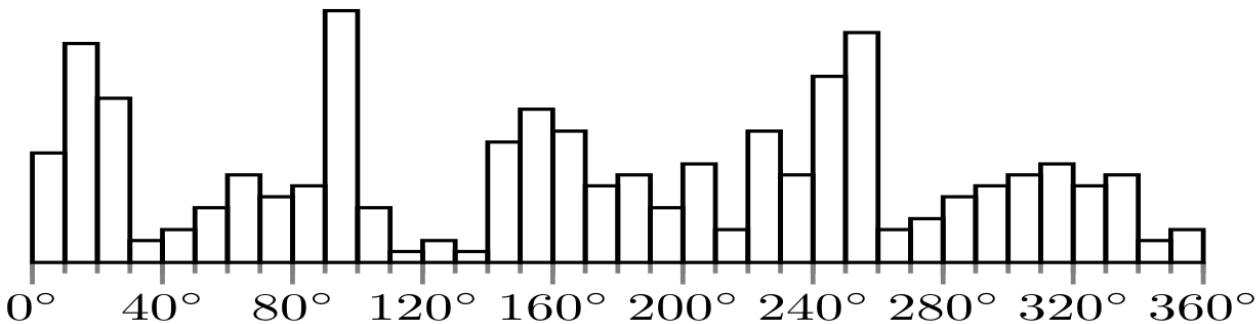
Histogram of Gradients (HoG)

- created for each feature point $\hat{\mathbf{x}}_i = (\hat{x}_i, \hat{y}_i, \hat{\sigma}_i)$.
- computed from gradients at the corresponding scale $\hat{\sigma}_i$.
- histogram uses 36 bins (orientations), each bin covering 10 degrees
- each histogram entry is weighted by the magnitude of the gradient ...
- ... and with a Gaussian centered at $(\hat{x}_i, \hat{y}_i)$ with standard deviation $1.5\hat{\sigma}_i$ that restricts computation to a neighbourhood.

Histogram of Gradients (HoG)



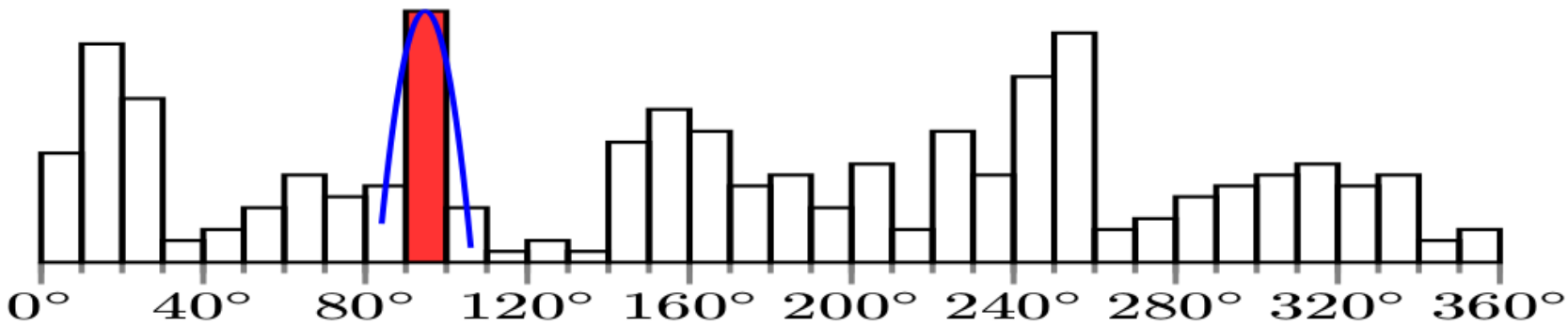




Top Left: computed gradients within a circular neighbourhood. The blue circle denotes three times the standard deviation of the Gaussian used as weighting function. **Top Right:** orientation decomposition in 36 bins (10 degrees per bin). **Bottom:** histogram of gradients.

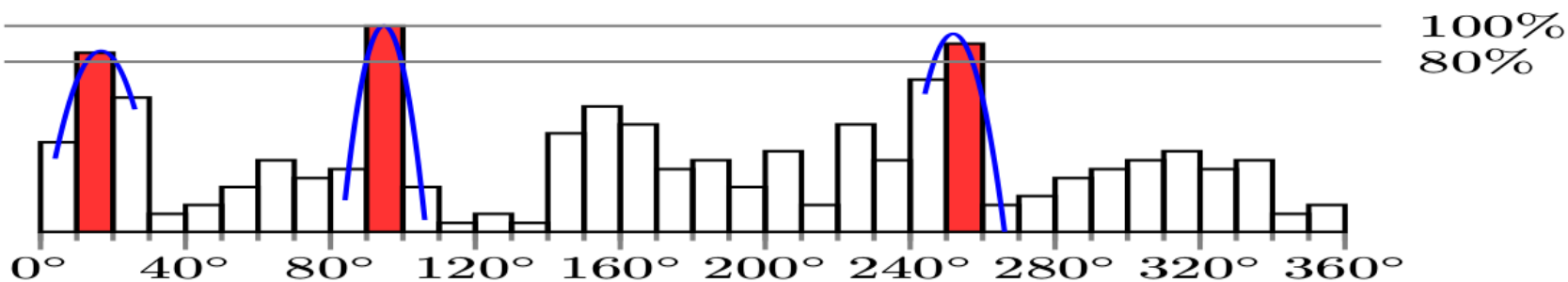
Estimation of the dominant orientation

- largest entry of histogram yields dominant orientation.
- sub-bin refinement is applied to improve the accuracy of the estimation (as in case of feature detection by fitting a quadratic function; however, this time only a 1D fit is required).

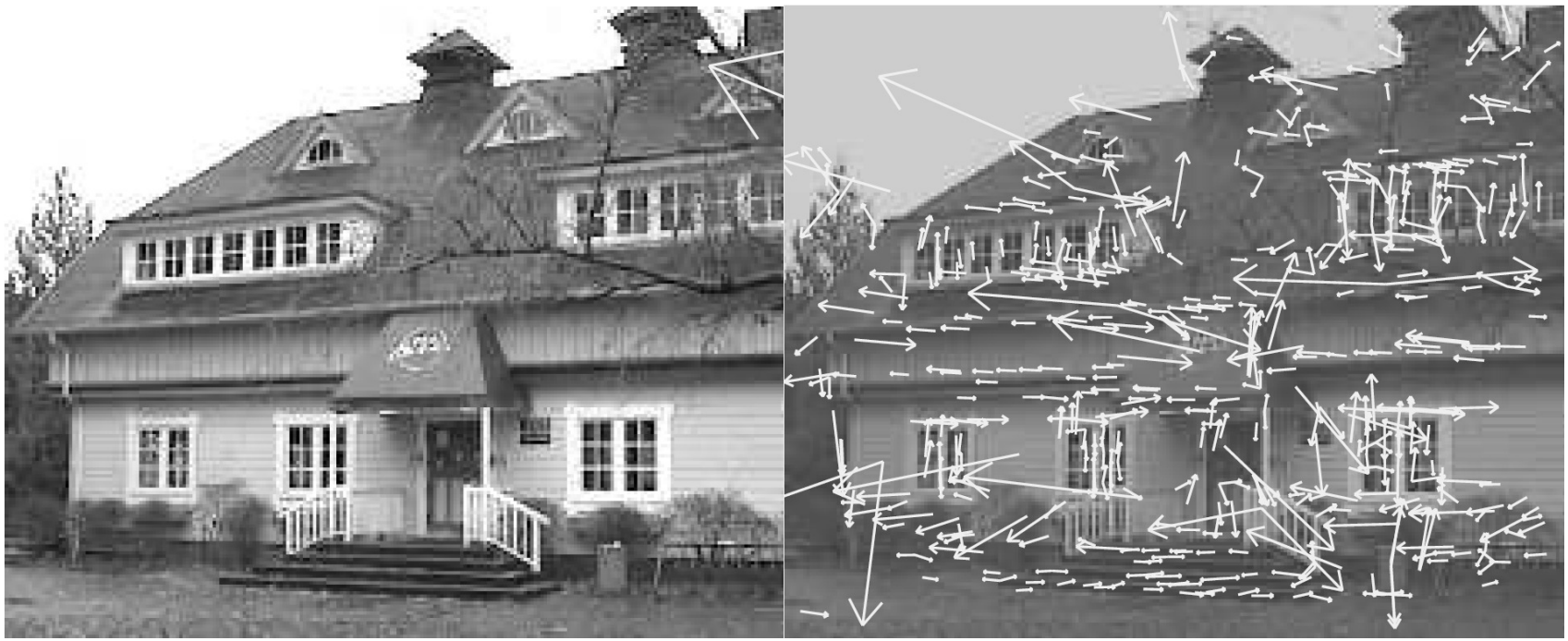


Feature cloning for multiple dominant orientations

- A separate feature point is created for each orientation with a histogram entry that is above 80% of the value of the dominant orientation.
- also in this case sub-bin refinement is applied.



Example 1: orientation assignment



Left: Test image of a house. **Right:** Detected features. The position of vectors represent the location of features, the magnitude their scale, the direction their orientation.

Example 2: orientation assignment

A more useful way to visualize features and orientation

Image created with VLfeat, http://www.vlfeat.org/

Overview

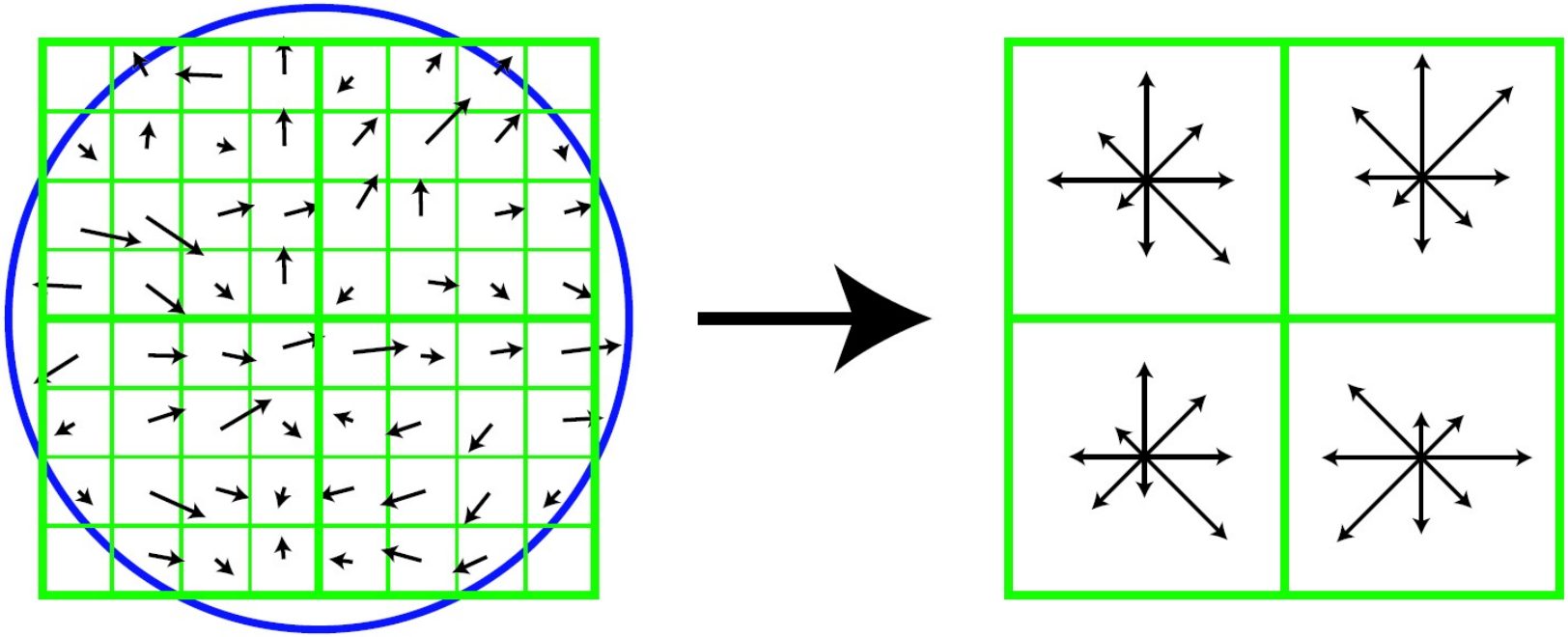
Step 4: computation of a suitable key point descriptor

How can we represent each feature point
(uniquely) by a characteristic vector ?

Block histogram of gradients

- start as in the case of the orientation assignment (consideration of scale and location, magnitude weighting, Gaussian weighting)
- in addition: sampling locations and gradient direction **compensated for dominant orientation**, image patch is rotated such that dominant orientation points into default direction (e.g. up).
- neighbourhood divided into 16 blocks of size 4×4 each
- histogram for each block uses only 8 bins each covering 45 degrees (less accuracy, but increased robustness)

Block histogram of gradients



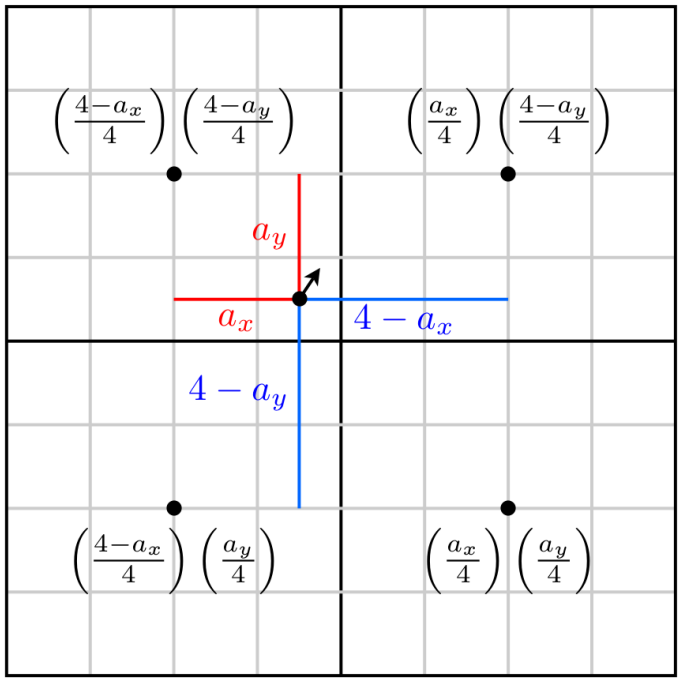
Example with four blocks of size 4×4. In an actual SIFT implementation, we have 16 instead of four blocks. The blue circle denotes three times the standard deviation of the Gaussian used as weighting function. **Left:** Computed gradients. **Right:** Histograms of gradients.

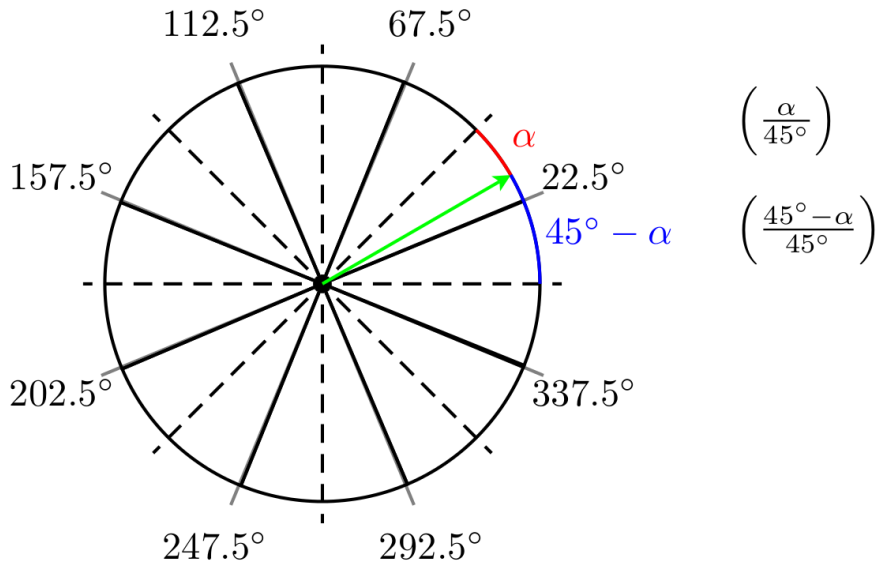
Drawback of discrete blocks and bins

- **Problem:** slight changes of the image in terms of rotations or translations may yield contributions of the same gradient to different bins and blocks, respectively.

- **Solution:** distribute votes of each gradient to the two neighbouring bins and four neighbouring blocks based on the angular and spatial distance.

Distribution of gradient sample to blocks and bins





Left: computed weights for the four neighbouring blocks (location). **Right:** computed weights for the two neighbouring bins (angle).

The SIFT descriptor

- stored for each feature point
- vector of size 128
- obtained by concatenating the 8 entries of all 16 block histograms normalised by its magnitude (unit vector)
- for matching, feature descriptors are compared by computing the Euclidean norm of their difference.

Example: features and visualization of the descriptors

Why is SIFT so successful ?

- stored neighbourhood information is computed at the characteristic scale relative to the feature location and compensated by the dominant orientation.

  → **invariance under scalings, shift and rotations**

- gradient information is invariant under additive illumination changes, while the final normalisation makes it invariant under multiplicative illumination changes.

  → **invariance under affine illumination changes**

- great engineering effort, extraordinatily robust also to more general illumination and viewpoint changes

  → **up to 60 degrees out-of-plane rotation demonstrated to still work quite well**

Overview

Summary

- The scale invariant feature transform (SIFT) allows to detect characteristic feature points in images.
- It is based on a Gaussian scale space using the difference of Gaussians (DoG).
- Detected extrema in scale space provide location and scale.
- The dominant orientation is found by a histogram of gradients.
- Location, scale and orientation are refined by fitting quadratic polynomials.
- Additional robustness is obtained by rejecting points with low contrast and high ratio of the principle curvatures.
- The final key point descriptor considers neighbourhood information and is based on block histogram of gradients.
- It is invariant with respect to changes in location, scale and orientation, and demonstrated to be extraordinarily robust with respect to more general changes.

References

- D. G. Lowe, *Object recognition from local scale-invariant features.* International Conference on Computer Vision, pp. 1150-1157, 1999.
  first paper on SIFT, focusses on object recognition

- D. G. Lowe, *Distinctive image features from scale-invariant keypoints.* International Journal of Computer Vision, Vol. 60(2), pp. 91-110, 2004.
  more detailed introduction to SIFT, this chapter

- R. Szeliski, *Computer Vision: Algorithms and Applications.* Feature selection and matching is covered in Chapter 4.1.