Object Recognition

Chapter 5: Local Features

Prof. Dr. Johannes Maucher

HdM CSM

Version 1.1
20. Mai 2017

Document History

Version Nr.	Date	Changes
1.0		Initial Version
1.1	29.04.2015	Anpassungen für SS 15

Chapter 3: Local Features

Local Features: SIFT Features

Example

- 422 keypoints found in image
- Each keypoint is described by a numeric vector of lentgh 128.

Central questions in this lecture

- What are keypoints?
- How to find keypoints?
- How to describe keypoints?

Purpose

Local Features as defined in [Grauman and Leibe, 2011]

The purpose of local invariant features is to provide a representation that allows to efficiently match local structures between images. That is, we want to obtain a sparse set of local measurements that capture the essence of the underlying input images and that encode their interesting structure.

Image Source: `http://cs.brown.edu/courses/cs143/results/proj2/sphene/`

Definition

Local Features as defined in [Tuytelaars and Mikolajczyk, 2007]

A local feature is an image pattern which differs from its immediate neighborhood. It is usually associated with a change of an image property or several properties simultaneously,... The image properties commonly considered are intensity, color, and texture.... Local features can be points, but also edges or small image patches. Typically, some measurements are taken from a region centered on a local feature and converted into descriptors. The descriptors can then be used for various applications.

I

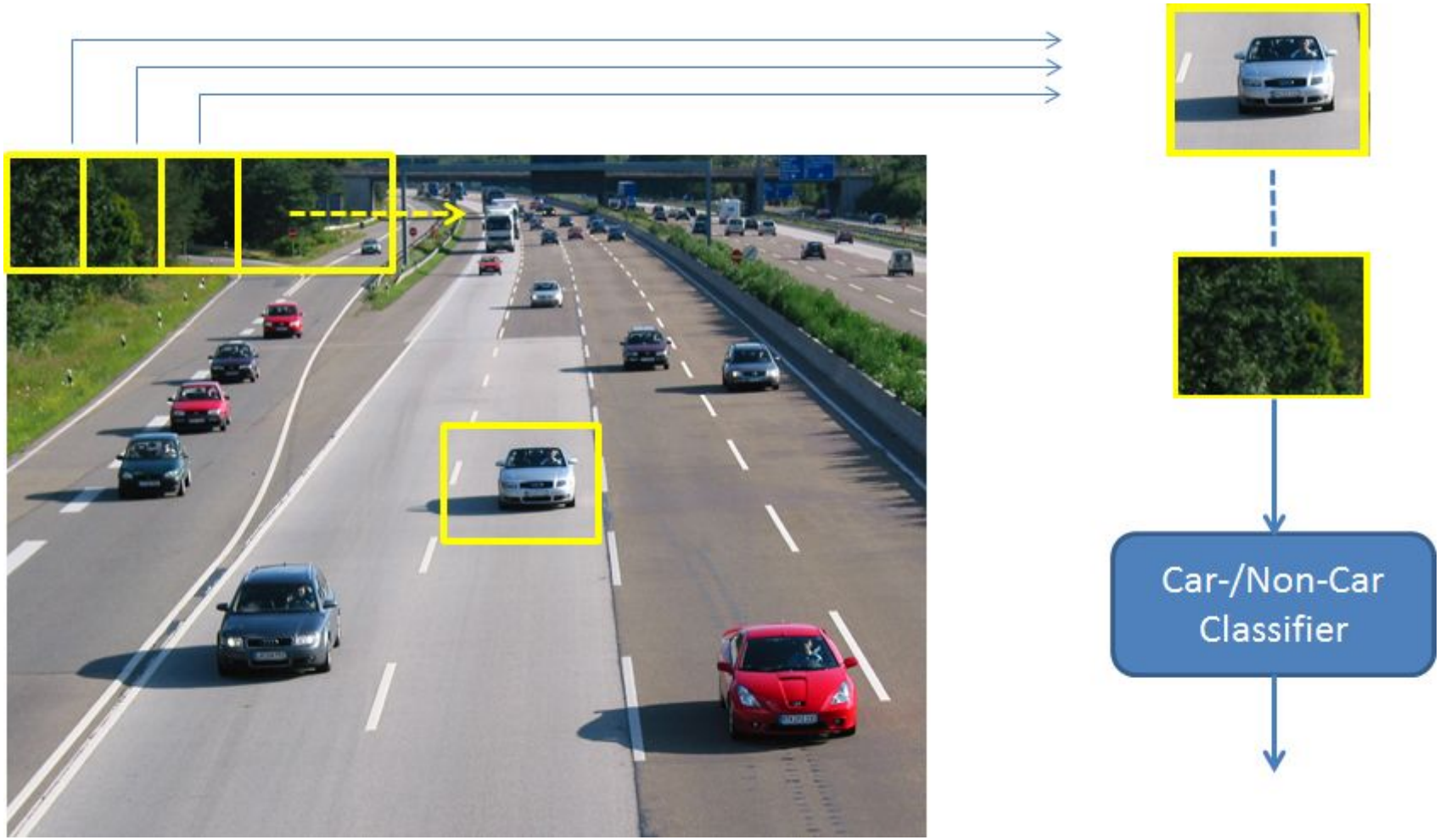
Local versus Global Features

- Global Features:
  - are calculated from the entire image, not only a local region
  - E.g. color histograms
  - Used e.g. for image retrieval
  - Cannot distinguish foreground and background or multiple objects in the image
  - In object recognition: E.g. Eigenface approach for face recognition uses global feature, but requires unique object/background separation or cropped images.
  - In general image segmentation can be applied to crop, but this is very challenging itself.
- Sliding Window Approach
  - calculates *Global Features* for each rectangular image partition,
  - works well for detection of predefined objects, e.g. pedestrians or cars.
  - cannot cope with partial occlusions
  - context information from outside the window is lost
  - deformable objects cannot captured well, e.g. animals in different poses.
  - very exhaustive calculation, since window must me slided over entire image.

Global Features Require Cropped Images

Source: `http://www.cl.cam.ac.uk/research/dtg/attarchive/facesataglance.html`

Global Features: Sliding Window Approach for Object Detection



Requirements for Good Features (1) ([Tuytelaars and Mikolajczyk, 2007])

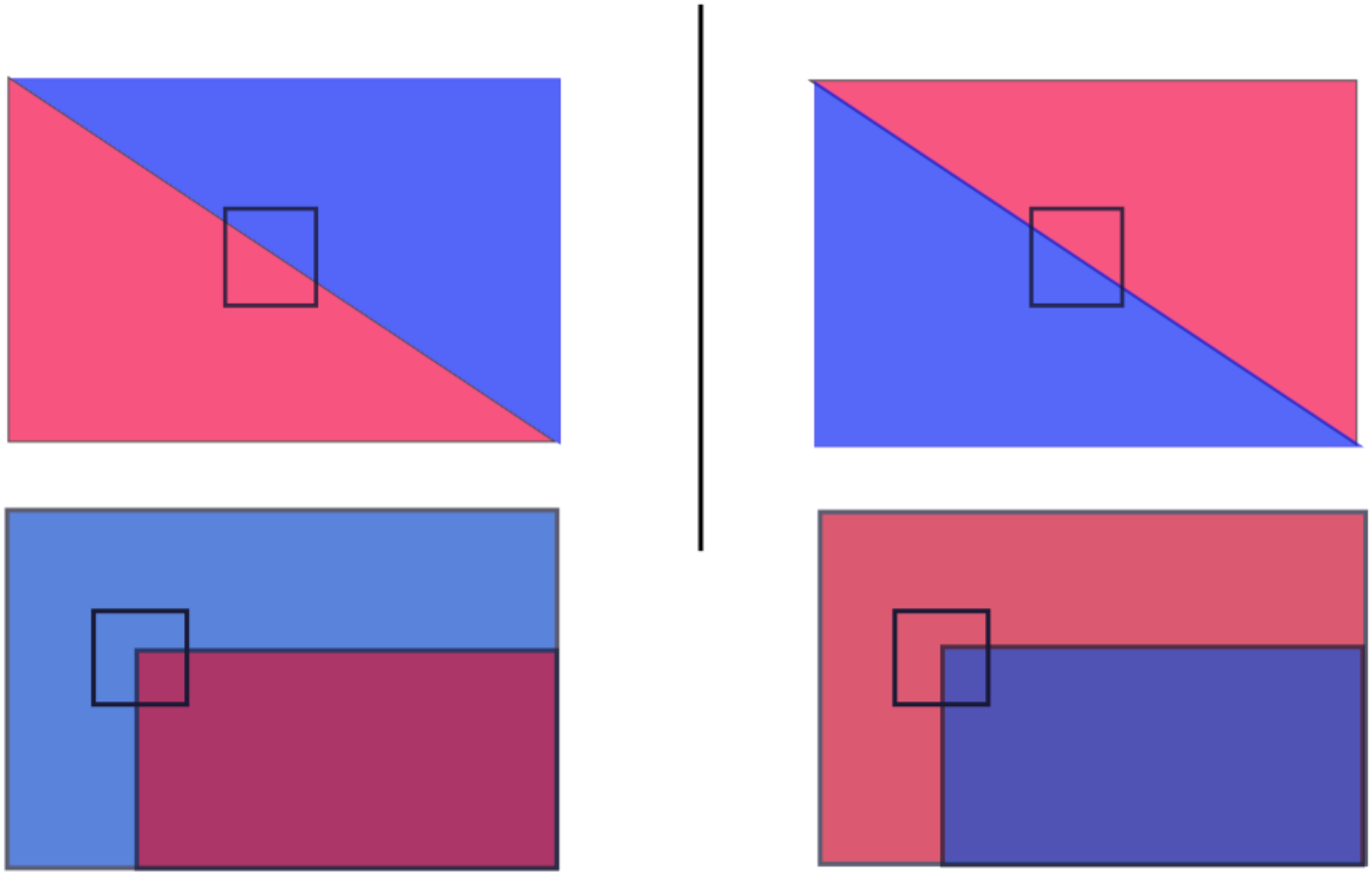
- Repeatability: Given two images of the same object, taken under different viewing conditions, a high percentage of the features detected on the object part visible in both images should be found in both images.
- Distinctiveness/informativeness: The intensity patterns underlying the detected features should show a lot of variation, such that features can be distinguished and matched.
- Locality: The features should be local, so as to reduce the probability of occlusion and to allow simple model approximations of the geometric and photometric deformations between two images taken under different viewing conditions.

Requirements for Good Features (2) ([Tuytelaars and Mikolajczyk, 2007])

- Quantity: The number of detected features should be sufficiently large, such that a reasonable number of features are detected even on small objects. However, the optimal number of features depends on the application. Ideally, the number of detected features should be adaptable over a large range by a simple and intuitive threshold. The density of features should reflect the information content of the image to provide a compact image representation.
- Accuracy: The detected features should be accurately localized, both in image location, as with respect to scale and possibly shape.
- Efficiency: Preferably, the detection of features in a new image should allow for time-critical applications.

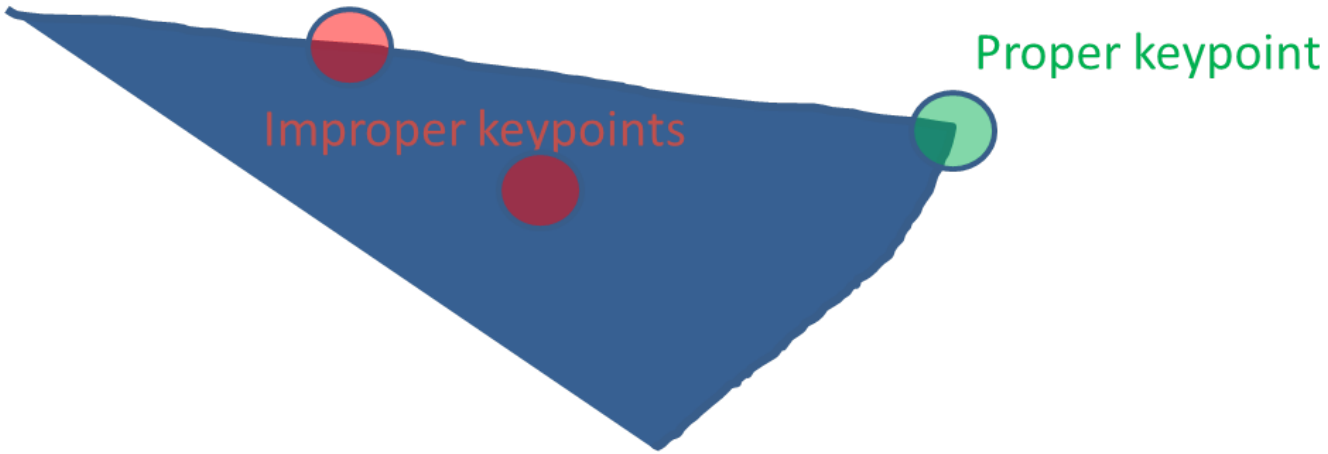
Requirement for Good Features - Distinctiveness(3)

Different Things should have different descriptors

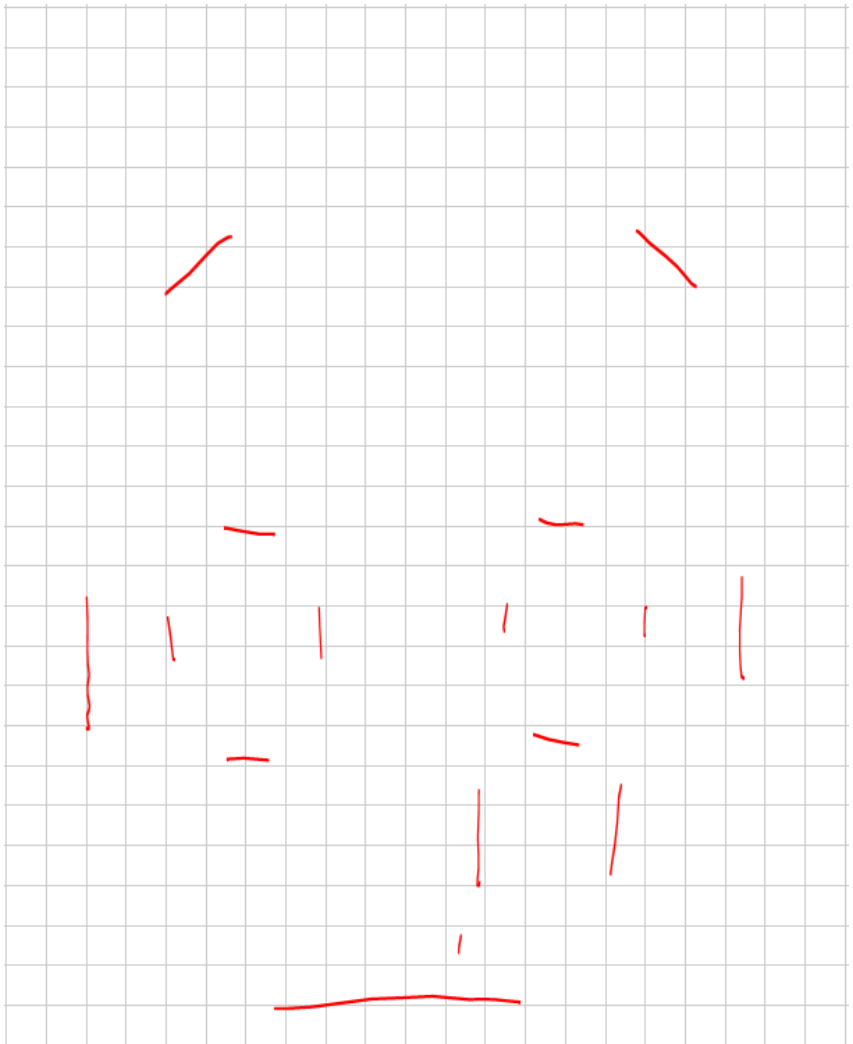


Requirement for Good Features - Accuracy (4)

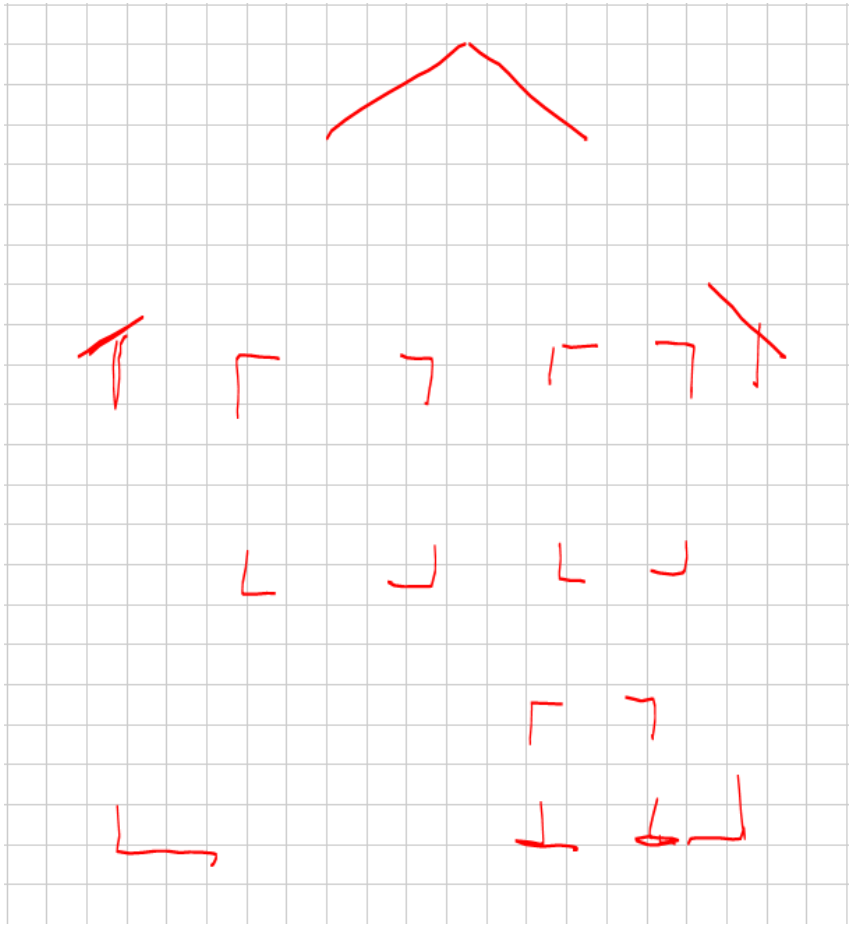
- By looking through a small window it must be easy to localize the point.
- Shifting the window in any direction should give a large change in pixel intensities, gradients, ...



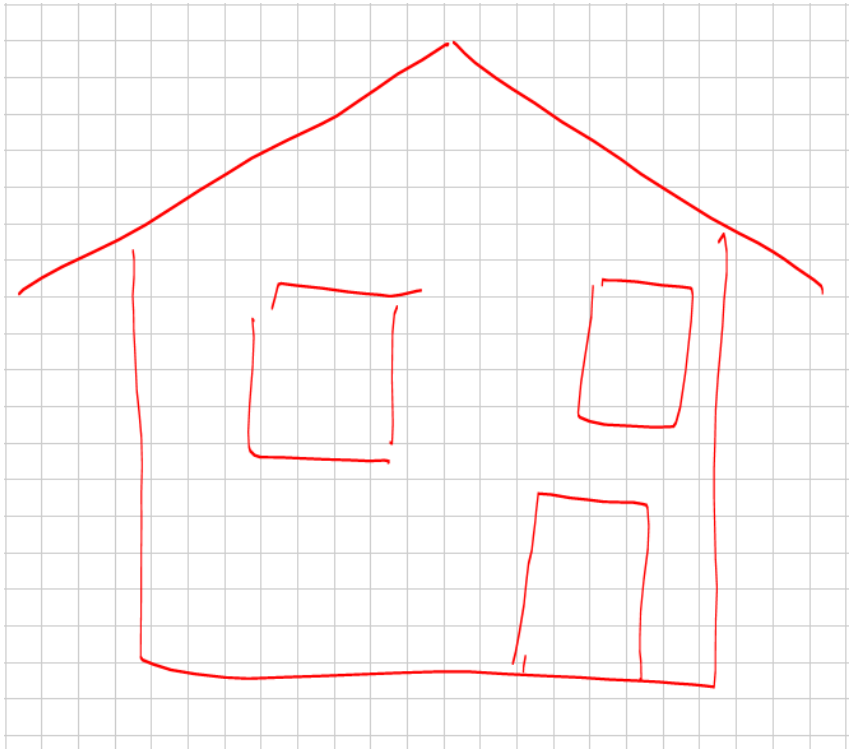
Good Features for the Human Brain - Whats this?



Good Features for the Human Brain - Whats this?



Good Features for the Human Brain - Thats it!



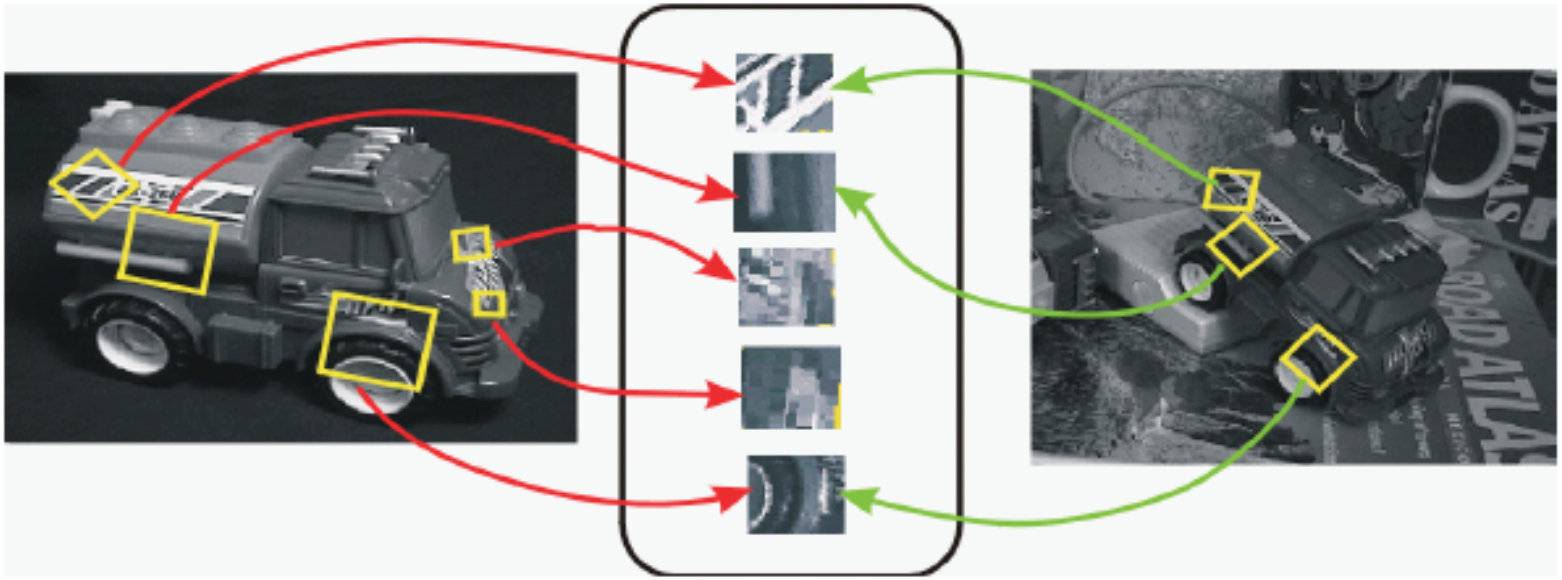
Categorization of Local Feature Applications ([Tuytelaars and Mikolajczyk, 2007])

1. Interest in a specific type of local feature that has a clear semantic interpretation in the context of a given application. E.g. in aerial images edges indicate roads.
2. Interest in local features since they provide a limited set of well localized and individually identifiable anchor points (e.g. for tracking applications or image stichting).
3. Use set of local features as a robust image representation. Allows for object- or scene recognition without image segmentation.

In the second and third category the semantics of the local feature are not relevant.

Local Feature based Recognition

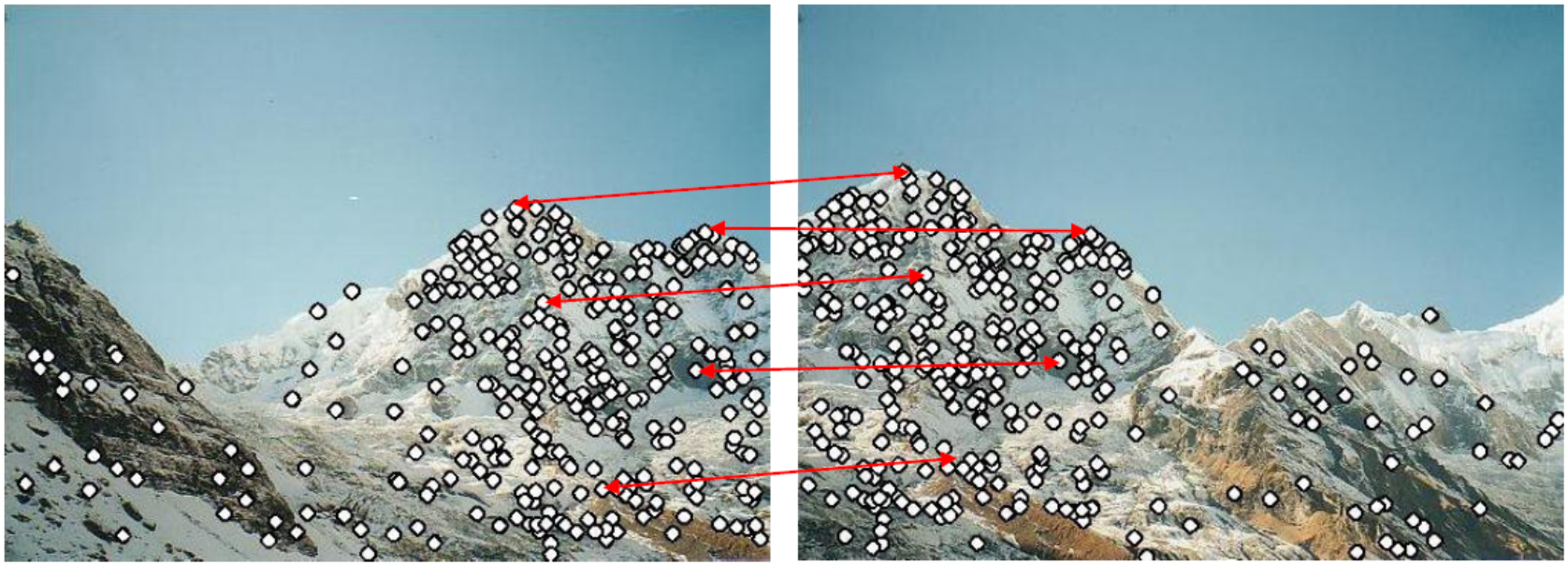
1. Extract local features from both images independently
2. Match the descriptors of the features to obtain candidate matching pairs
3. Verify if features occur in a consistent geometric configuration



Quelle: [Grauman and Leibe, 2011]

Local Features based Image Stitching

Goal: Align multiple image to generate one panorama view.



- Detect keypoints in both images
- Find pairs of corresponding keypoints
- Use the location of corresponding keypoints ...

Local Features based Image Stitching

- ... for aligning the images

Harris/Förstner Corner Detection (1)

- Criteria for good keypoints as stated on slide 1: Shifting the small window in any direction should give a large change in gradients.
- According to [Harris and Stephens, 1988] the change of gradients in the neighborhood of a pixel can be determined as follows:

1. Calculate the gradient at position $\mathbf{x}$:

$$\nabla I(\mathbf{x}) = \begin{pmatrix} \frac{\partial I(\mathbf{x})}{\partial x} \\ \frac{\partial I(\mathbf{x})}{\partial y} \end{pmatrix} = \begin{pmatrix} I_x(\mathbf{x}) \\ I_y(\mathbf{x}) \end{pmatrix} \tag{1}$$

   The gradients are calculated by applying the 1st order derivative of a Gaussian (see chapter Image Processing).

2. At each position $\mathbf{x}$ calculate the matrix

$$M_I(\mathbf{x}) = \nabla I(\mathbf{x}) \nabla I^T(\mathbf{x}) = \begin{pmatrix} I_x(\mathbf{x}) \\ I_y(\mathbf{x}) \end{pmatrix} (I_x(\mathbf{x}) \quad I_y(\mathbf{x})) = \begin{pmatrix} I_x^2(\mathbf{x}) & I_x I_y(\mathbf{x}) \\ I_x I_y(\mathbf{x}) & I_y^2(\mathbf{x}) \end{pmatrix} \tag{2}$$

Harris Corner Harris/Förstner Corner Detection (2)

3. Average matrices $M_l(\mathbf{x})$ over a region by convolution with a Gaussian G_σ:

$$C(\mathbf{x}, \sigma) = G_\sigma(\mathbf{x}) * M_l(\mathbf{x}) \tag{3}$$

4. Depending on the local image properties in the region defined by the width of G_σ the Eigenvalues λ_1 and λ_2 of matrix $C(\mathbf{x}, \sigma)$ vary as follows:
   - If both Eigenvalues are large, then there is a corner around **x**
   - If one Eigenvalue is large, and the other is ≈ 0, then there is an edge around **x**.
   - If both Eigenvalues are ≈ 0, then there is no variation in the region around **x**.

Detecting Edges without calculating Eigenvalues (3)

- Denote:
  - α: eigenvalue with larger magnitude
  - β: eigenvalue with smaller magnitude
- A small ratio
$$r = \frac{\alpha}{\beta}$$
indicates a corner.
- According to [Harris and Stephens, 1988] the ratio of eigenvalues can be determined in a very efficient manner, without actually calculating the eigenvalues:
  - Calculate trace and determinant[1] of $C(\mathbf{x}, \sigma)$
$$Tr(\mathbf{C}(\mathbf{x}, \sigma)) = \alpha + \beta \quad (4)$$
$$Det(\mathbf{C}(\mathbf{x}, \sigma)) = \alpha\beta \quad (5)$$
then
$$\frac{Tr(\mathbf{C}(\mathbf{x}, \sigma))^2}{Det(\mathbf{C}(\mathbf{x}, \sigma))} = \frac{(\alpha + \beta)^2}{\alpha\beta} = \frac{(r\beta + \beta)^2}{r\beta^2} = \frac{(r+1)^2}{r} \quad (6)$$
depends only on the ratio r of eigenvalues.
- Corners are at the points where this value is small.

[1] http://en.wikipedia.org/wiki/Trace_(linear_algebra)

Detecting Edges without calculating Eigenvalues (4)

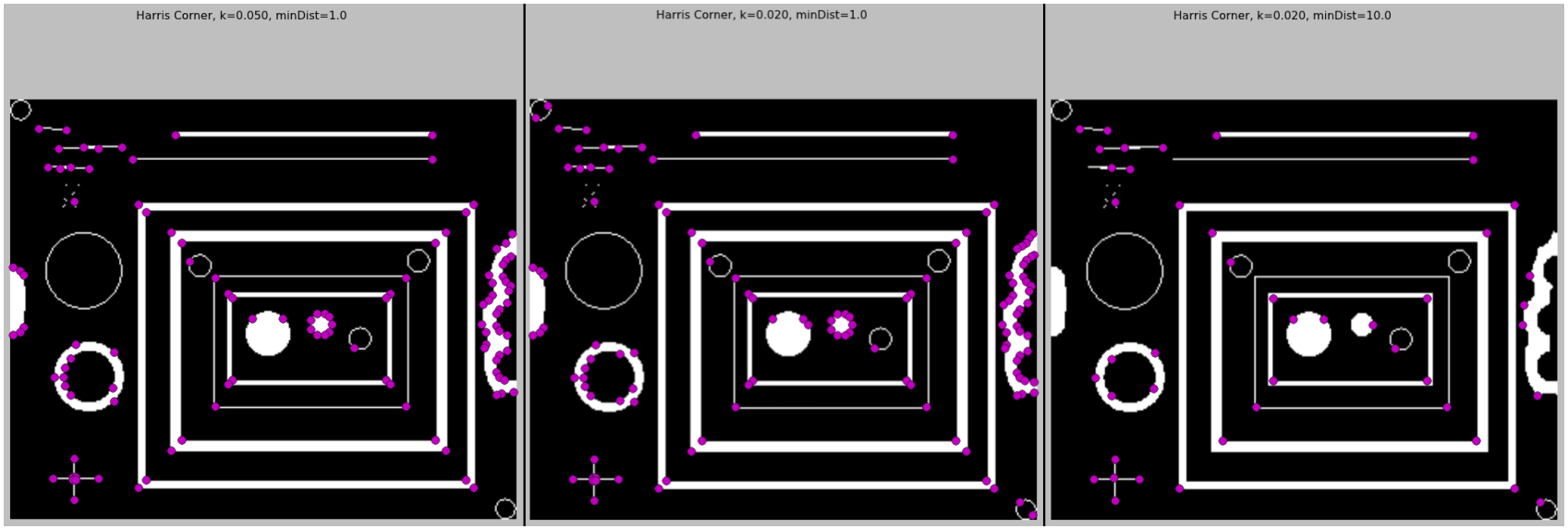
- In order to check if the value in equation (6) is small one can also check if

$$Det(\mathbf{C}(\mathbf{x}, \sigma)) - \kappa Tr(\mathbf{C}(\mathbf{x}, \sigma))^2 > T \tag{7}$$

- Remarks on the parameters:
  - κ is in the range $[0.04, \ldots 0.15]$. Smaller κ yields more detected corners.
  - If the standard deviation of the Gaussian filter, used to calculate the derivatives in equation (1) is σ_d, then the standard deviation σ of the Gaussian, used in equation (3), should be $\sigma \approx 2\sigma_d$. Typical value: $\sigma = 1.0$ i.e. $\sigma_d \approx 0.5$.
- In implementations, e.g. Scikits Image[2], usually
  - one function calculates for all pixels the value, given in the left hand side of unequation (7) and
  - a second function applies the threshold T (default value 0) and a minimum distance D_{min} to the result of the first function. Detected corners are then separated by at least D_{min} pixels.

[2] http://scikit-image.org/docs/dev/api/api.html

Harris Corner Detection: Varying κ and D_{min}



Harris Corner Detection: Properties

- Results are well suited for finding stereo correspondences
- Rotation invariant
- Not scale invariant

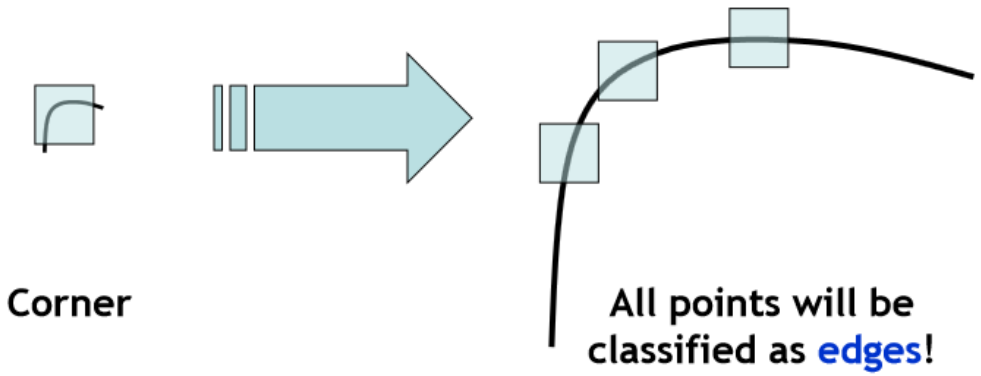


Figure: Source: Kristen Graumann

Scale Invariant Feature Transform (SIFT): Stages

1. Scale-space extrema detection: Search potential keypoints by determining extremas in scale space.
2. Keypoint localization
3. Orientation assignment: One or more orientations are assigned to each keypoint location.
4. Keypoint descriptor: Local image gradients are measured at the selected scale in the region around each keypoint.

Characteristics

Invariant w.r.t.

- translation
- rotation
- scale
- illumination (partially invariant)
- 3D viewpoint
- additional noise

Highly distinctive: For each single feature the correct match in the database features is found with high probability

Scale Space

- Scale Space of input image $I(x, y)$:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{8}$$

- Variable Scale Gaussian:

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2 + y^2}{2\sigma^2}} \tag{9}$$

- Difference of Gaussian (DoG):

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma). \end{aligned} \tag{10}$$

- DoG is
  - efficient to compute - the smoothed images L must be computed in any case for the scale space feature description
  - a close approximation to the scale-normalized Laplacian of Gaussian, whose extremas produce the most stable image features ([Lowe, 2004])

Scale Space and Difference of Gaussians

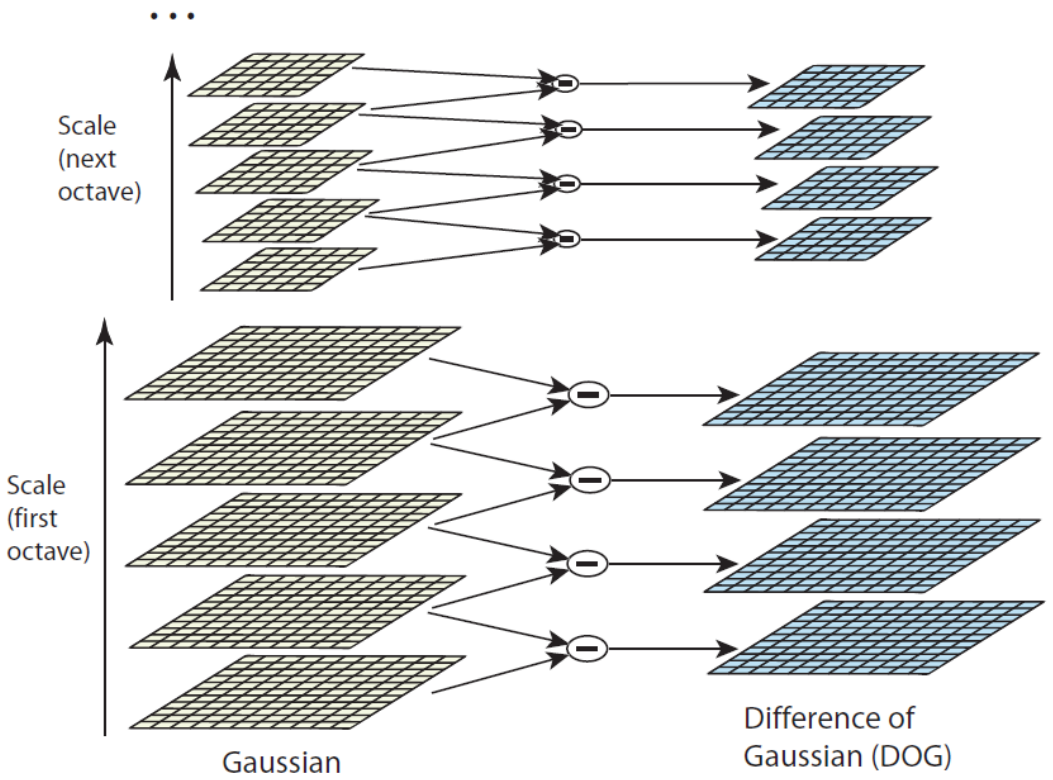
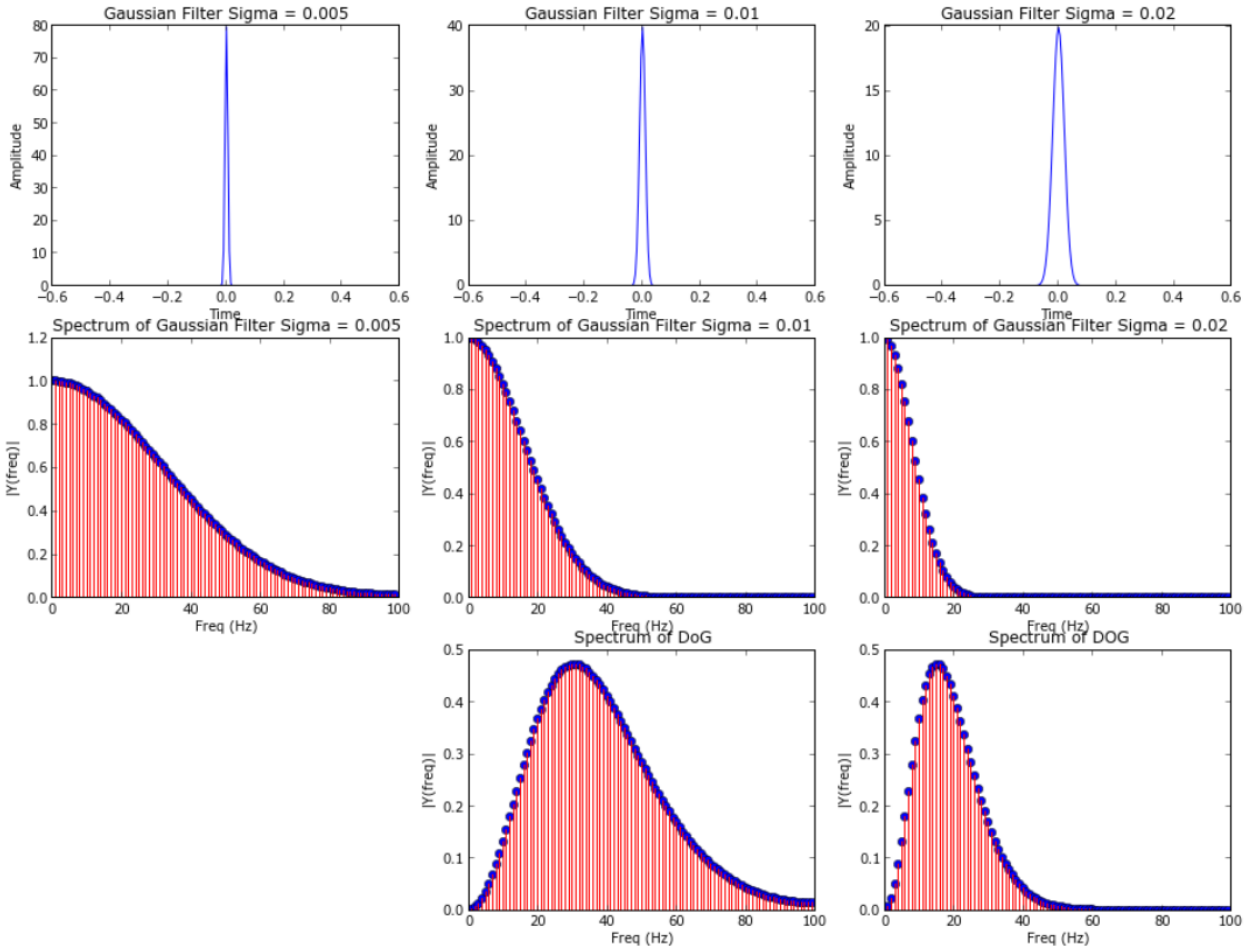


Figure 1: For each octave of scale space, the initial image is repeatedly convolved with Gaussians to produce the set of scale space images shown on the left. Adjacent Gaussian images are subtracted to produce the difference-of-Gaussian images on the right. After each octave, the Gaussian image is down-sampled by a factor of 2, and the process repeated.

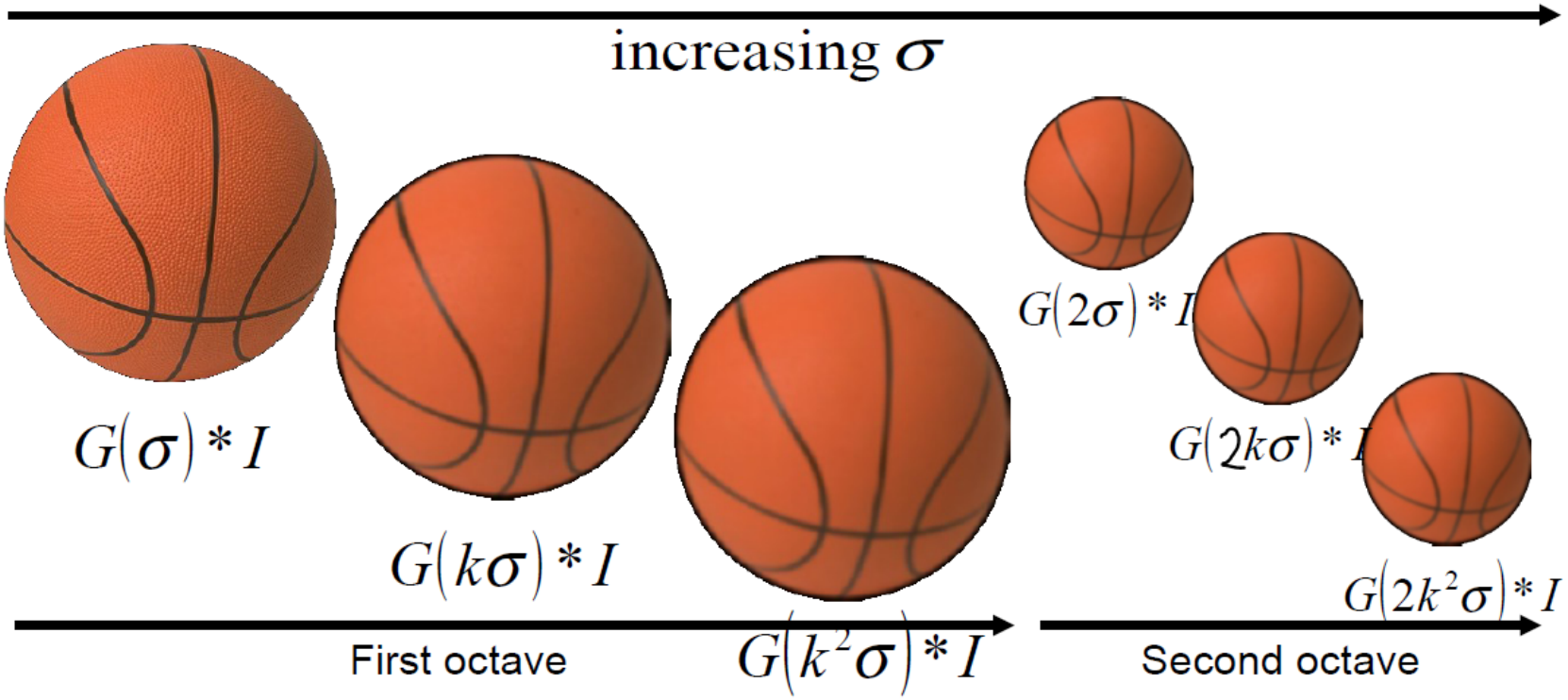
- Octave: Doubling of σ
- Scale space resolution: $k = 2^{1/s}$, where s is the number of scales per octave.
- After each octave the image is subsampled by a factor of 2.
- $s + 3$ blurred images per octave are required to determine scale space extrema (see next slide)
- Typical value: $s = 3$

Source: [Lowe, 2004]

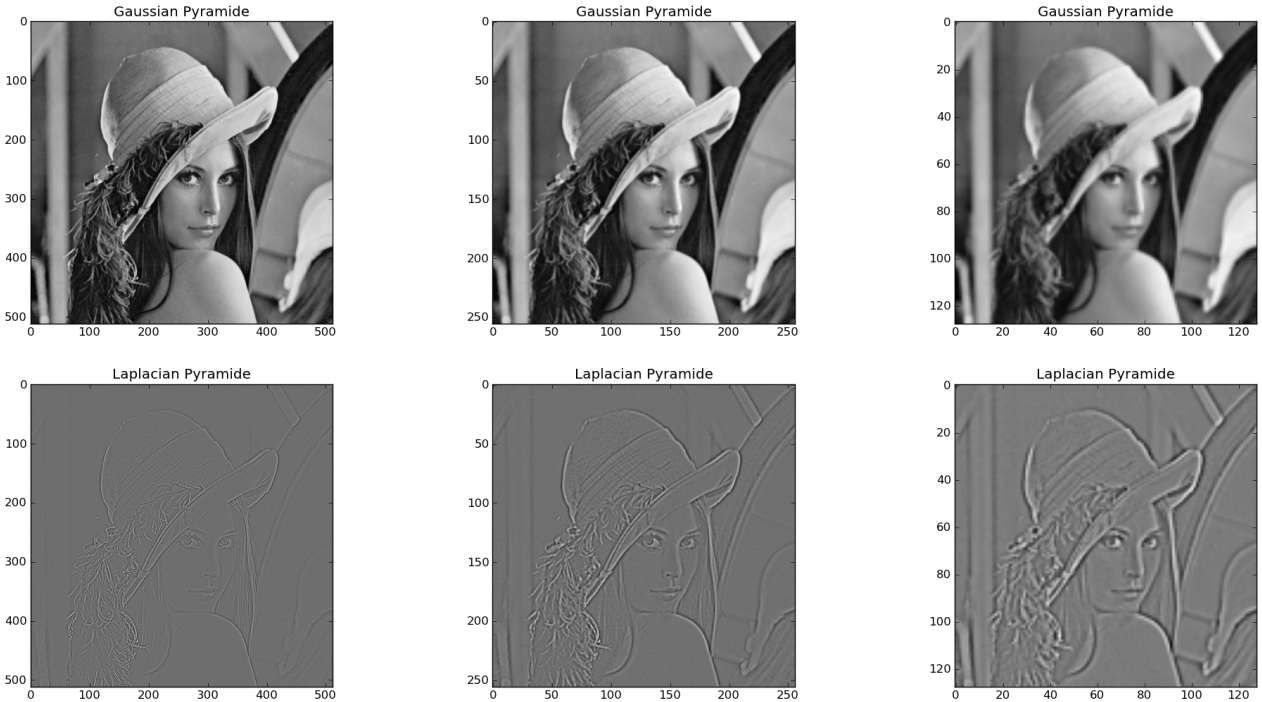
Spectrum of Gaussian- and Laplacian Pyramide



Gaussian Pyramide Example



Difference of Gaussian (=Laplacian) Example



Laplacian in first column is difference between the unfiltered original image and the first Gaussian filtered image ($\sigma = 0.66$)

Extrema Detection in DoG Scale Space

- Each Pixel in the DoG image is compared with its 26 neighbour pixels at the current and adjacent scales
- The Pixel is selected as a keypoint candidate, if it is larger than all of the neighbours (maxima) or smaller than all of them (minima).

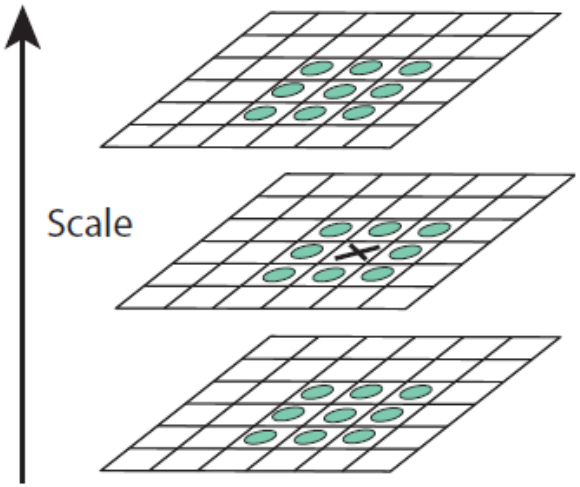


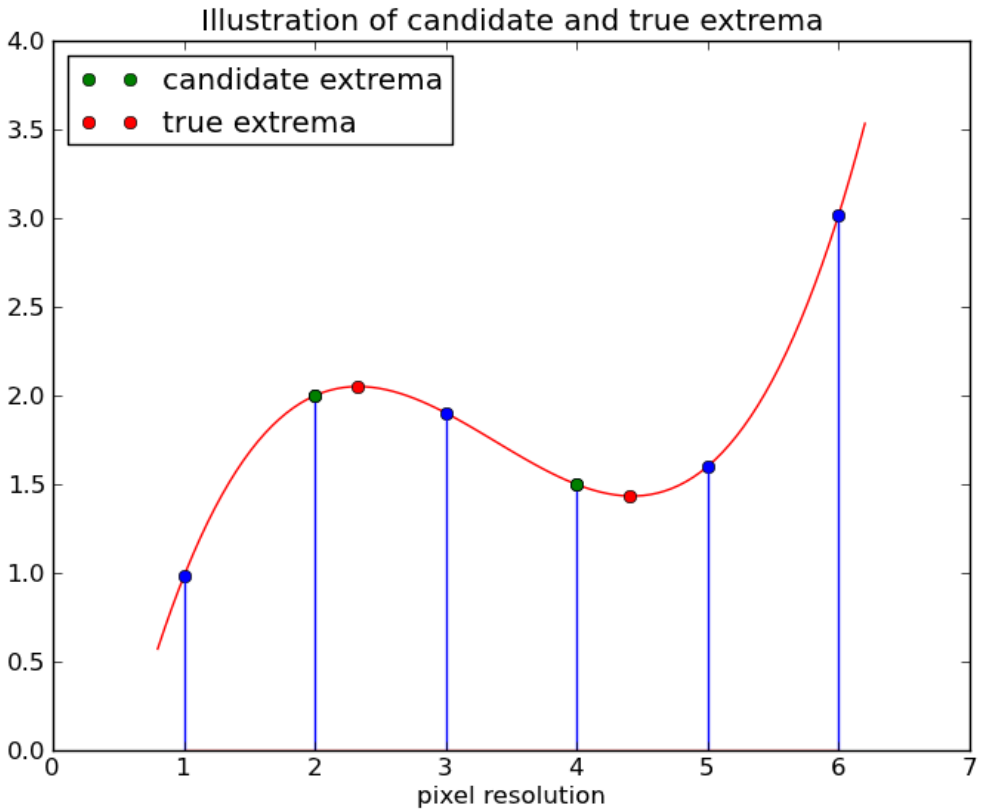
Figure 2: Maxima and minima of the difference-of-Gaussian images are detected by comparing a pixel (marked with X) to its 26 neighbors in 3x3 regions at the current and adjacent scales (marked with circles).

Source: [Lowe, 2004]

Goals of Accurate Keypoint Localization

- Determine exact position of local extremas on sub-pixel resolution
- Select within the set of candidate keypoints only the stable and distinctive ones.
- Candidate keypoints with low contrast are not stable, because they are sensitive to noise
- Candidate keypoints on edges are not distinctive

Interpolation for Keypoint Localization



- Candidate keypoints are detected only on pixel-resolution
- At higher scales, this corresponds to several pixels in base image
- Localization of true extremas on sub-pixel resolution by interpolation

Fitting a 3D Quadratic Function to Local Sample Points

- Taylor expansion of $D(x, y, \sigma)$ up to quadratic forms around the sample point:

$$D(\mathbf{x}) = D + \frac{\partial D^T}{\partial \mathbf{x}}\mathbf{x} + \frac{1}{2}\mathbf{x}^T\frac{\partial^2 D}{\partial \mathbf{x}^2}\mathbf{x}. \tag{11}$$

Here

- $\mathbf{x} = (x, y, \sigma)^T$ is the offset from the candidate keypoint.
- $$\frac{\partial D}{\partial \mathbf{x}} = \begin{pmatrix} \frac{\partial D}{\partial x} \\ \frac{\partial D}{\partial y} \\ \frac{\partial D}{\partial \sigma} \end{pmatrix} = \begin{pmatrix} D_x \\ D_y \\ D_\sigma \end{pmatrix}$$
- $$\frac{\partial^2 D}{\partial \mathbf{x}^2} = \begin{pmatrix} D_{xx} & D_{xy} & D_{x\sigma} \\ D_{yx} & D_{yy} & D_{y\sigma} \\ D_{\sigma x} & D_{\sigma y} & D_{\sigma\sigma} \end{pmatrix}$$

Fitting a 3D Quadratic Function to Local Sample Points

- Derivatives of D are approximated by using differences in the (3x3)-neighbourhood.
- True location $\hat{\mathbf{x}}$ of the keypoint is determined by taking the derivative of (11) and setting it to zero:

$$\hat{\mathbf{x}} = -\frac{\partial^2 D}{\partial \mathbf{x}^2}^{-1} \frac{\partial D}{\partial \mathbf{x}}. \tag{12}$$

- If the offset $\hat{\mathbf{x}}$ is > 0.5 in any dimension, then extremum lies closer to different sample point and new interpolation starts centered at this new point.

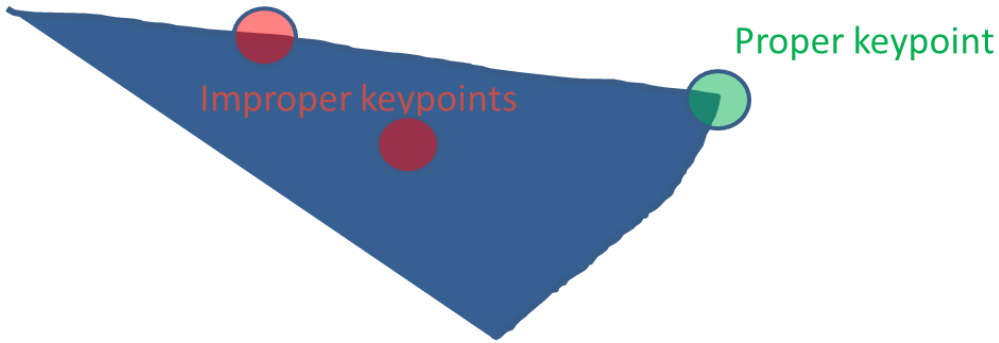
Rejecting Low Contrast Candidates

- Pixels of low contrast can be determined by low DoG values
- If the image values are normalized to the range $[0, 1]$, candidates with

$$|D(\hat{\mathbf{x}})| < 0.03$$

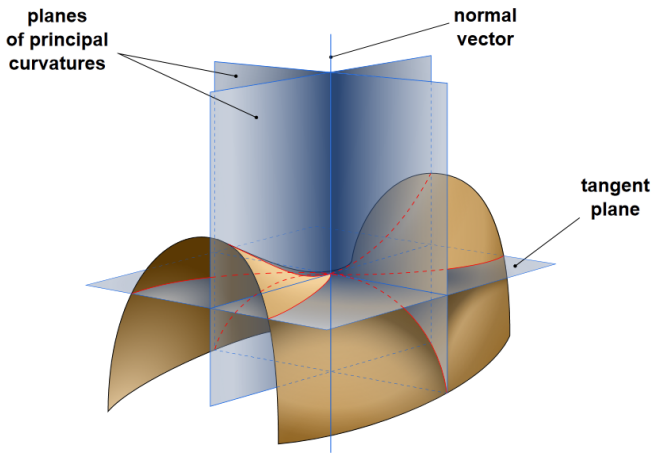
are rejected as low contrast points.

Rejecting Keypoint Candidates on Edges



- DoG values can not uniquely locate points on edges (since all keypoints on the edge have similar DoG values), but in corners.
- Edges constitute extremas in the DoG, therefore they are selected as candidates in the first step.
- At edges the DoG function will have a large prinicpal curvature across the edge, but a relatively small curvature in the orthogonal direction. This criteria is applied for rejecting edge points.

Principal Curvatures



Source: http://en.wikipedia.org/wiki/Principal_curvature

- The two eigenvalues of the 2D Hessian

$$\mathbf{H} = \begin{pmatrix} D_{xx} & D_{xy} \\ D_{yx} & D_{yy} \end{pmatrix} \tag{13}$$

are proportional to the principal curvatures.

Ratio of Principal Curvatures

- Denote:
  - α: eigenvalue with larger magnitude
  - β: eigenvalue with smaller magnitude
- A large ratio

$$r = \frac{\alpha}{\beta}$$

  indicates a large curvature in one direction and a small in the perpendicular direction $\Rightarrow$ Edge
  - Calculate trace and determinant of $\mathbf{H}$

$$Tr(\mathbf{H}) = D_{xx} + D_{yy} = \alpha + \beta \quad (14)$$

$$Det(\mathbf{H}) = D_{xx}D_{yy} - (D_{xy})^2 = \alpha\beta \quad (15)$$

    then

$$\frac{Tr(\mathbf{H})^2}{Det(\mathbf{H})} = \frac{(\alpha+\beta)^2}{\alpha\beta} = \frac{(r\beta+\beta)^2}{r\beta^2} = \frac{(r+1)^2}{r} \quad (16)$$

    depends only on the ratio r of eigenvalues.
- In [Lowe, 2004] candidates with $r > 10$ are rejected as edges.

Example: Keypoint Detection, Localization and Filtering ([Lowe, 2004])

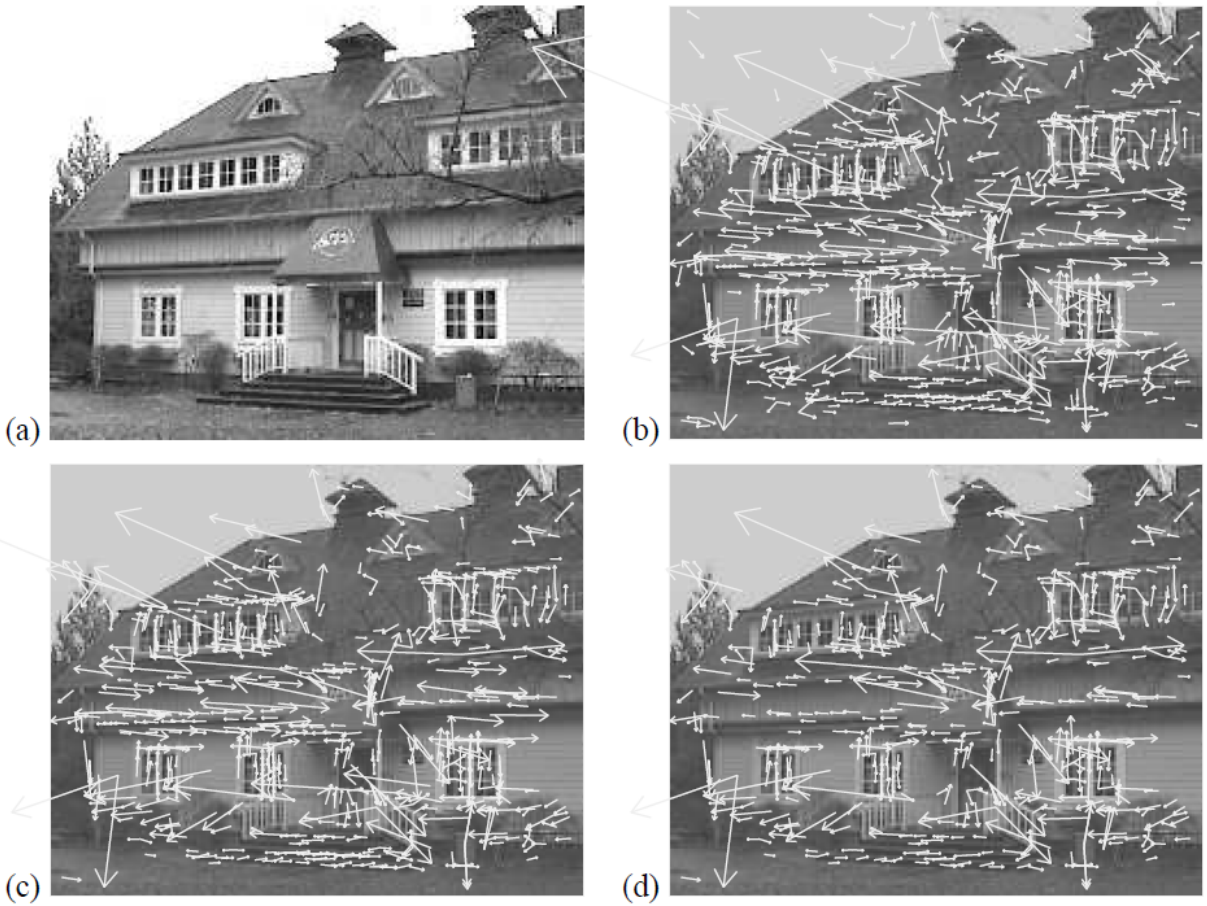


Figure 5: This figure shows the stages of keypoint selection. (a) The 233x189 pixel original image. (b) The initial 832 keypoints locations at maxima and minima of the difference-of-Gaussian function. Keypoints are displayed as vectors indicating scale, orientation, and location. (c) After applying a threshold on minimum contrast, 729 keypoints remain. (d) The final 536 keypoints that remain following an additional threshold on ratio of principal curvatures.

Orientation Assignment: Idea

- Calculate for each keypoint a consistent orientation, based on local image properties
- Describe each keypoint relative to this orientation
- $\Rightarrow$ Invariance w.r.t. image rotation.

Create Orientation Histogram (1)

1. From the scale of the keypoint, select the Gaussian smoothed image, L, with the closest scale.
2. For each image sample, $L(x, y)$ at the selected scale compute gradient magnitude

$$m(x, y) = \sqrt{(L(x+1, y) - L(x-1, y))^2 + (L(x, y+1) - L(x, y-1))^2} \tag{17}$$

and gradient orientation

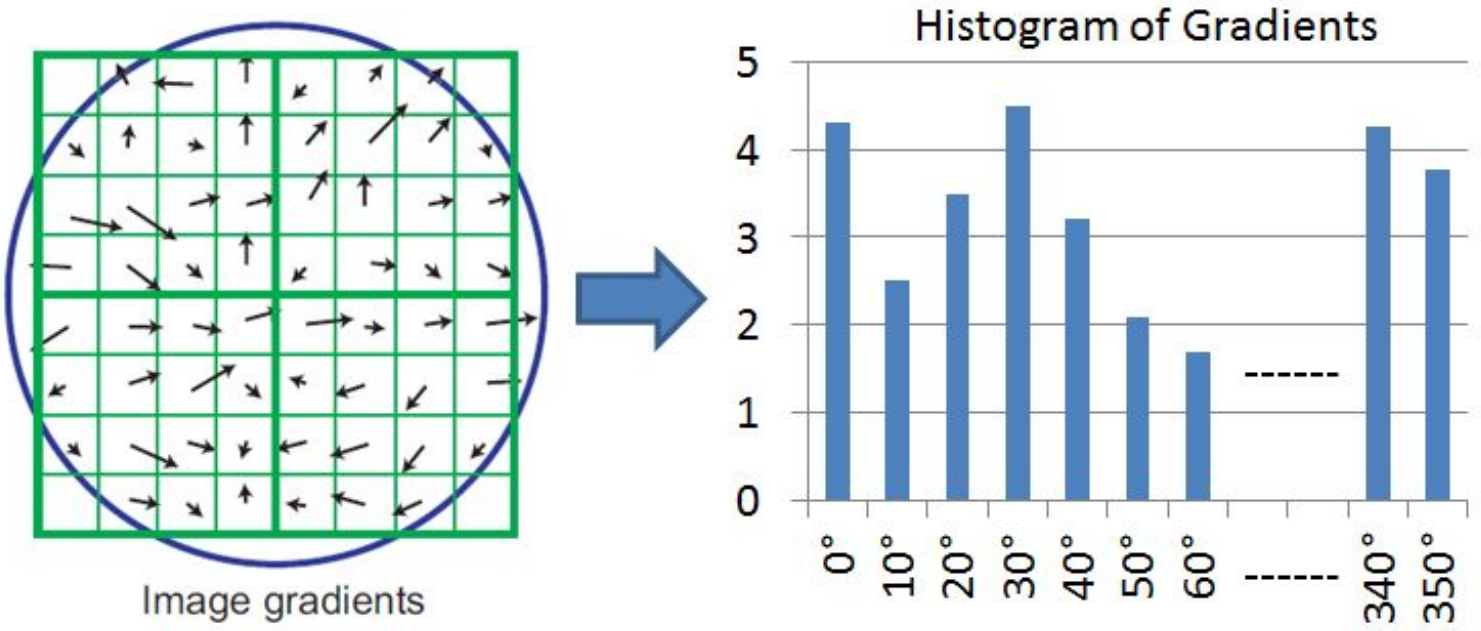
$$\theta(x, y) = \tanh \frac{L(x, y+1) - L(x, y-1)}{L(x+1, y) - L(x-1, y)} \tag{18}$$

3. Calculate a 36 bin gradient histogram ($1 bin \triangleq 10°$) from the gradients of the samples in the local neighbourhood of the keypoint.

Create Orientation Histogram (2)

The contribution of each gradient to the histogram is the product of

- its magnitude $m(x, y)$
- and the value $g(x', y')$, where $g(x', y')$ is a circular Gaussian window, centered at the keypoint (x, y). The standard deviation σ is 1.5 times the scale of the keypoint.



Assign Orientations to Keypoints

- For each histogram the orientations with the
  - peak value
  - with values $>$ 80% of the histogram's peak

  constitute keypoint orientations.
- Thus for a given keypoint location $>$ 1 keypoints can be generated.
- In the average 15% of all keypoints have multiple orientations.
- For better accuracy of the keypoint orientations: Quadratic interpolation w.r.t. the 3 closest histogram values.

Computation of Keypoint Descriptor

For each keypoint:

- Determine gradient magnitude and orientations at all sample points around the keypoint as described above.
- In order to achieve orientation invariance, the coordinates of the descriptor and the gradient orientations are rotated relative to the keypoint orientation.
- A Gaussian weighting function with σ equal to one half the width of the descriptor window is used to assign a weight to the magnitude of each sample point.
- Thus the influence of sample points at the window boundary is reduced and the keypoint descriptor is less sensitive to small shifts of the window.
- For each of the $(4x4)$-subregions a single histogram is created.
- Each histogram has 8 bins (directions).
- For a descriptor window of size $(16x16)$, there are 16 subregions and each keypoint descriptor is a vector of $4 \cdot 4 \cdot 8 = 128$ elements.

2x2 Descriptor

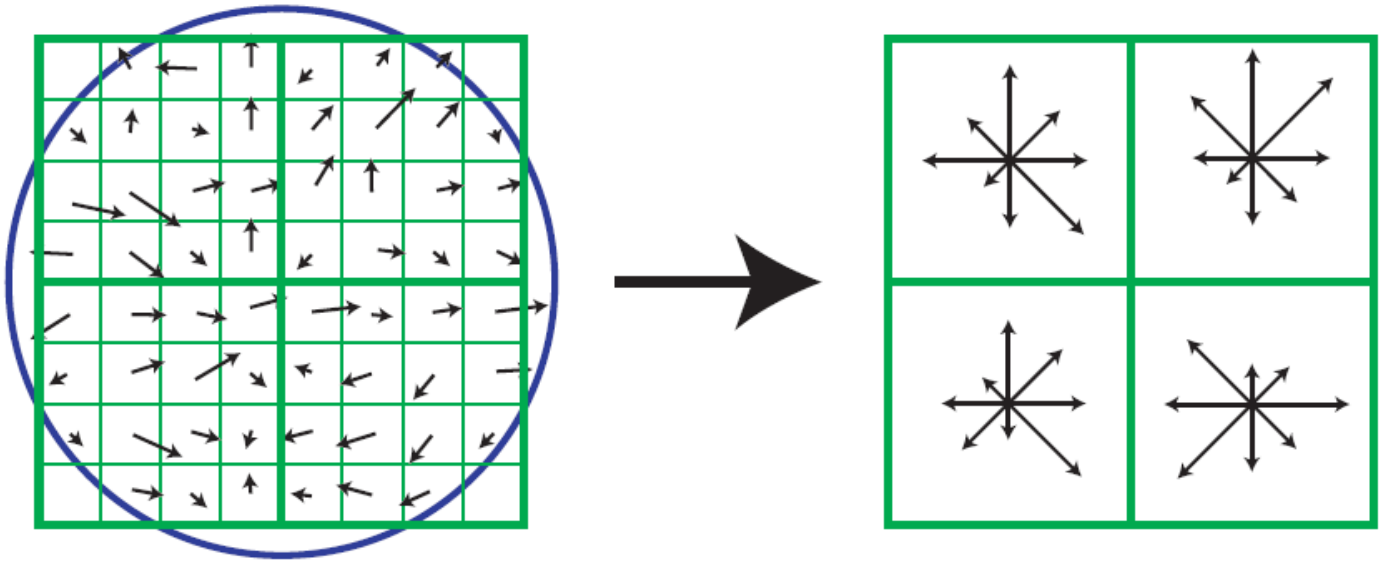
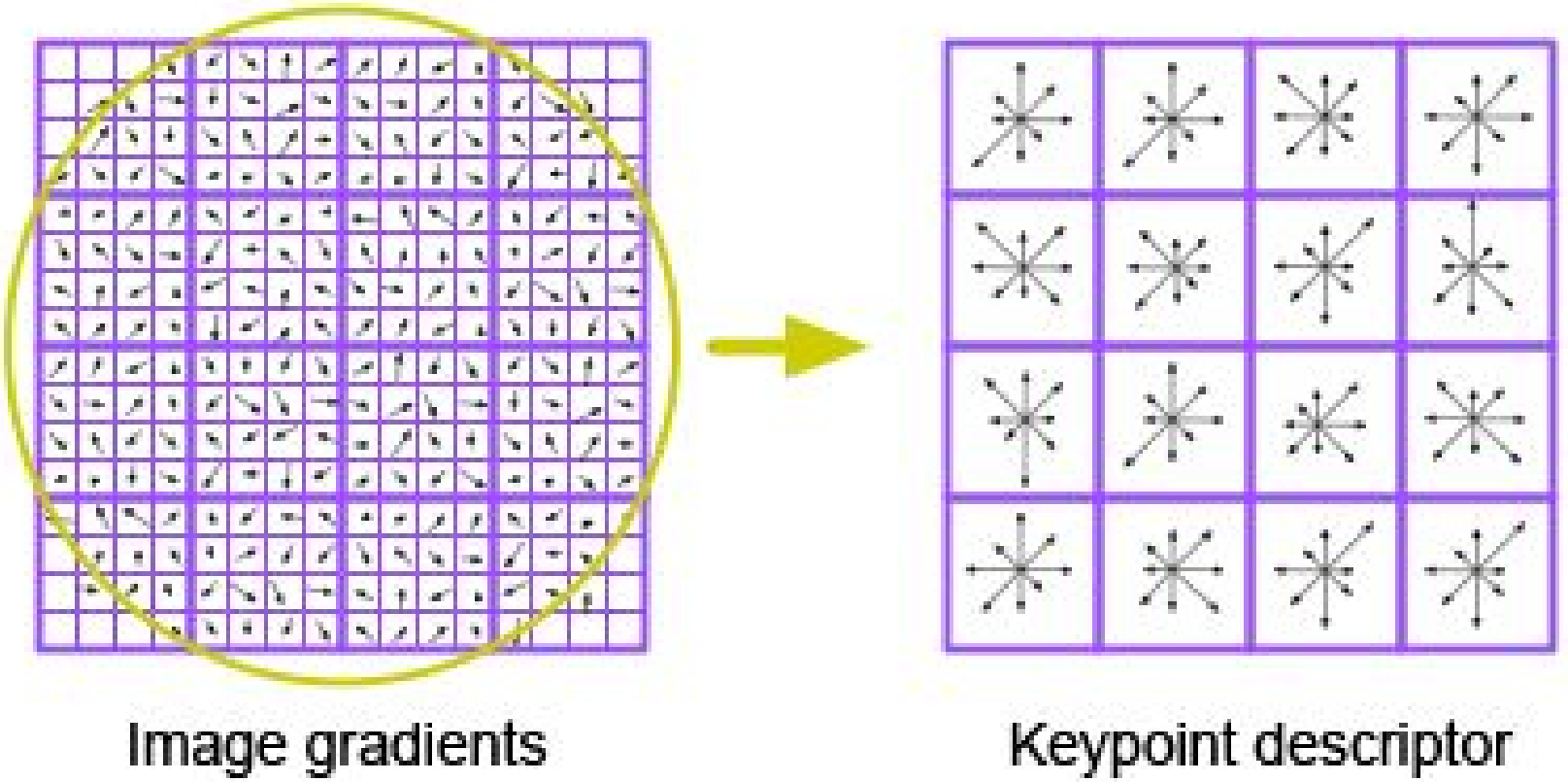


Figure 7: A keypoint descriptor is created by first computing the gradient magnitude and orientation at each image sample point in a region around the keypoint location, as shown on the left. These are weighted by a Gaussian window, indicated by the overlaid circle. These samples are then accumulated into orientation histograms summarizing the contents over 4x4 subregions, as shown on the right, with the length of each arrow corresponding to the sum of the gradient magnitudes near that direction within the region. This figure shows a 2x2 descriptor array computed from an 8x8 set of samples, whereas the experiments in this paper use 4x4 descriptors computed from a 16x16 sample array.

Source: [Lowe, 2004]

4x4 Descriptor



Source:
`http://www.cg.tu-berlin.de/fileadmin/fg144/Courses/06WS/scanning/Jonas/html/index.html`

Avoid Abrupt Descriptor Changes

- Descriptor may change abruptly if a sample shifts smoothly from
  - one histogram to another
  - one orientation bin to another
- Such abrupt changes are avoided by applying trilinear interpolation[3], which distributes the value of each gradient sample into adjacent histogram bins.

[3] http://en.wikipedia.org/wiki/Trilinear_interpolation

Reduce Illumination Change Effects

Contrast Change:

- Multiplication of all pixel values by a constant factor
- Gradients are multiplied by the same constant
- Contrast change can be canceled by normalizing each feature vector to unit length.

Brightness Change:

- Constant is added to all pixel values
- No effect on gradients, since difference remains the same.

$\Rightarrow$ Descriptor is invariant to affine illumination changes (but not to non-linear changes)

References I

Grauman, K. and Leibe, B. (2011).

Visual Object Recognition.

Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers.

Harris, C. and Stephens, M. (1988).

A combined corner and edge detector.

In *In Proc. of Fourth Alvey Vision Conference*, pages 147–151.

Lowe, D. G. (2004).

Distinctive image features from scale-invariant keypoints.

International Journal of Computer Vision, 60(2):91–110.

Tuytelaars, T. and Mikolajczyk, K. (2007).

Local invariant feature detectors: A survey.

Foundations and Trends in Computer Graphics and Vision, 3(3):177–280.