Speeded Up Robust Features (SURF)

Harris corner detector algorithm

- Compute magnitude of the gradient everywhere in x and y directions I_x, I_y
- Compute $I_x^2, I_y^2, I_x I_y$
- Convolve these three images with a Gaussian window, w. Find M for each pixel,

$$M = \sum w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

- Compute detector response, R at each pixel.

$$R = det(M) - k(trace(M))^2$$

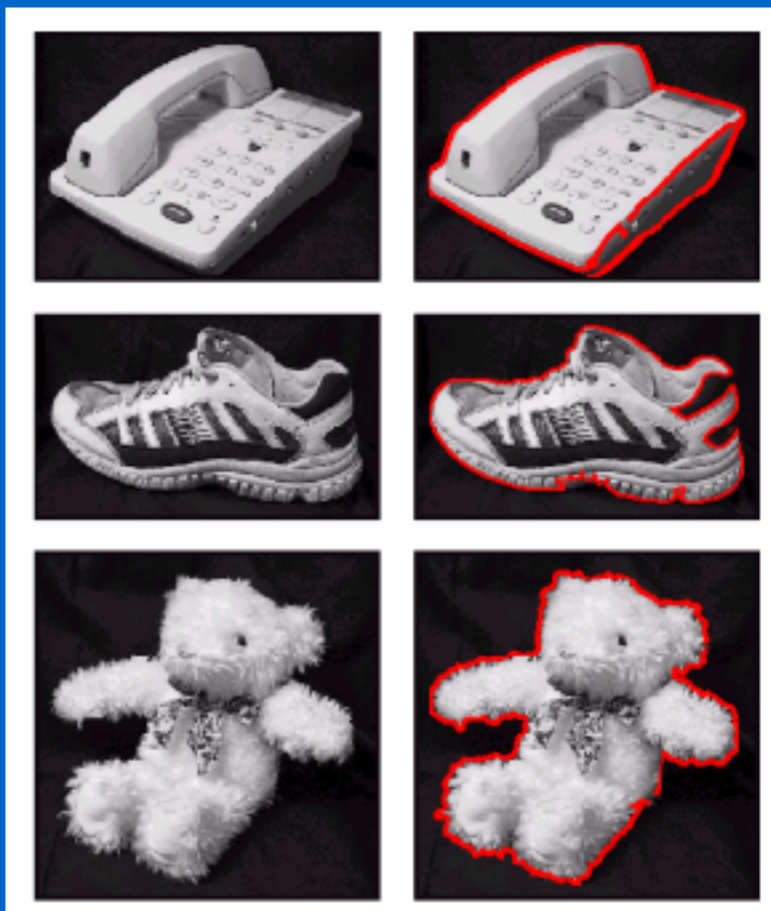
- find local maxima above some threshold on R. Compute nonmax suppression.

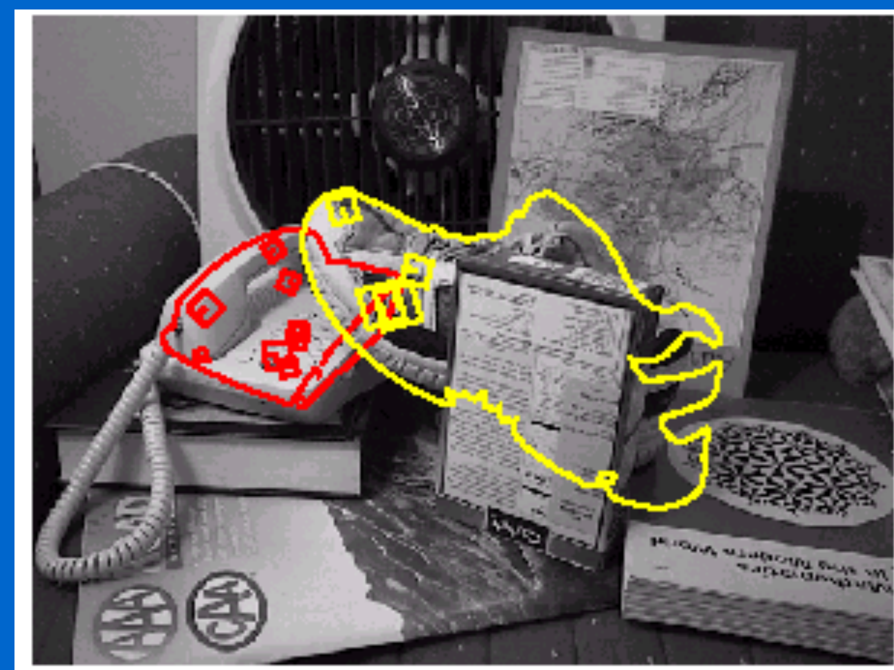
Applications of SIFT

- Object recognition
- Object categorization
- Location recognition
- Robot localization
- Image retrieval
- Image panoramas

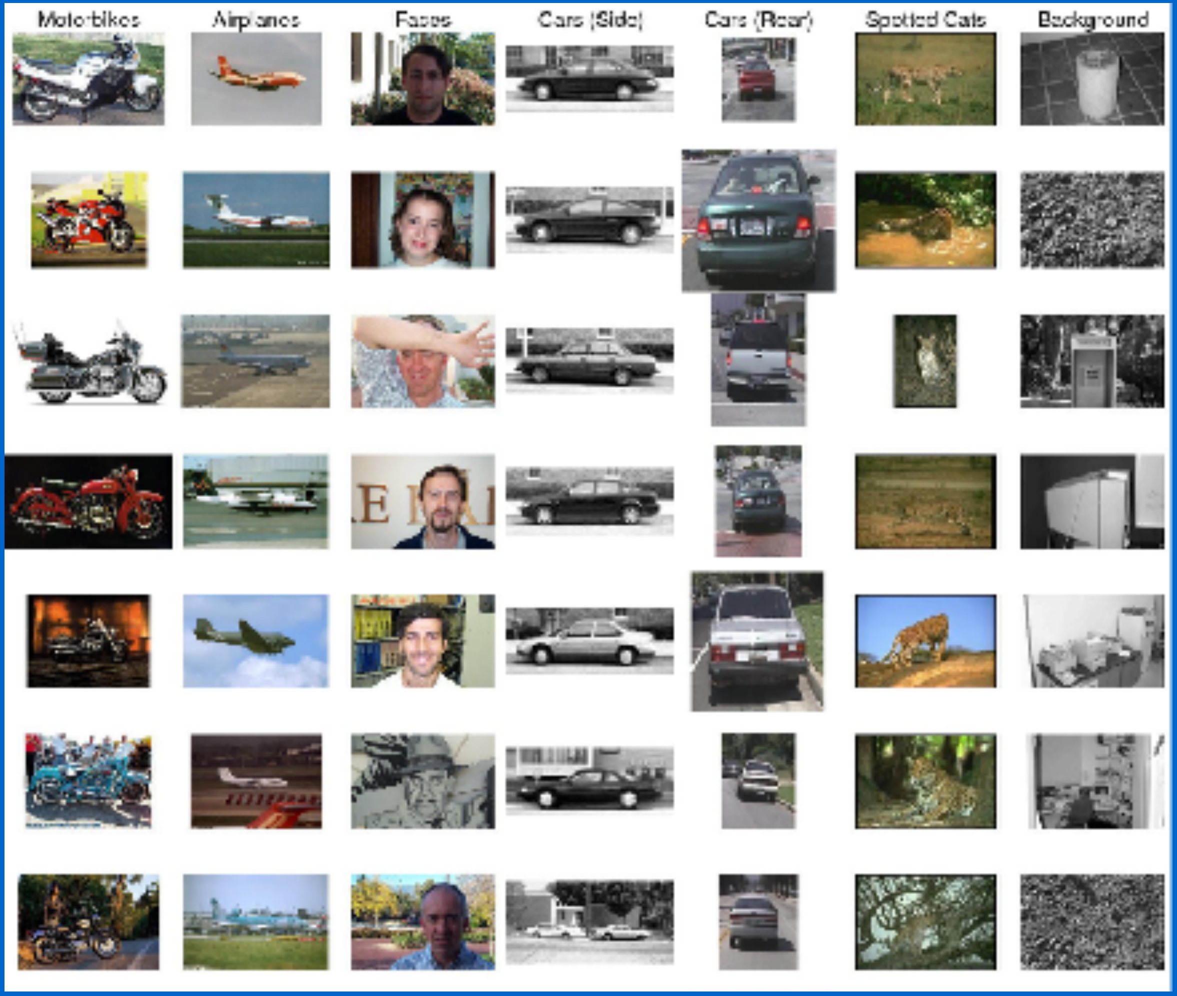
Object Recognition

Object Models





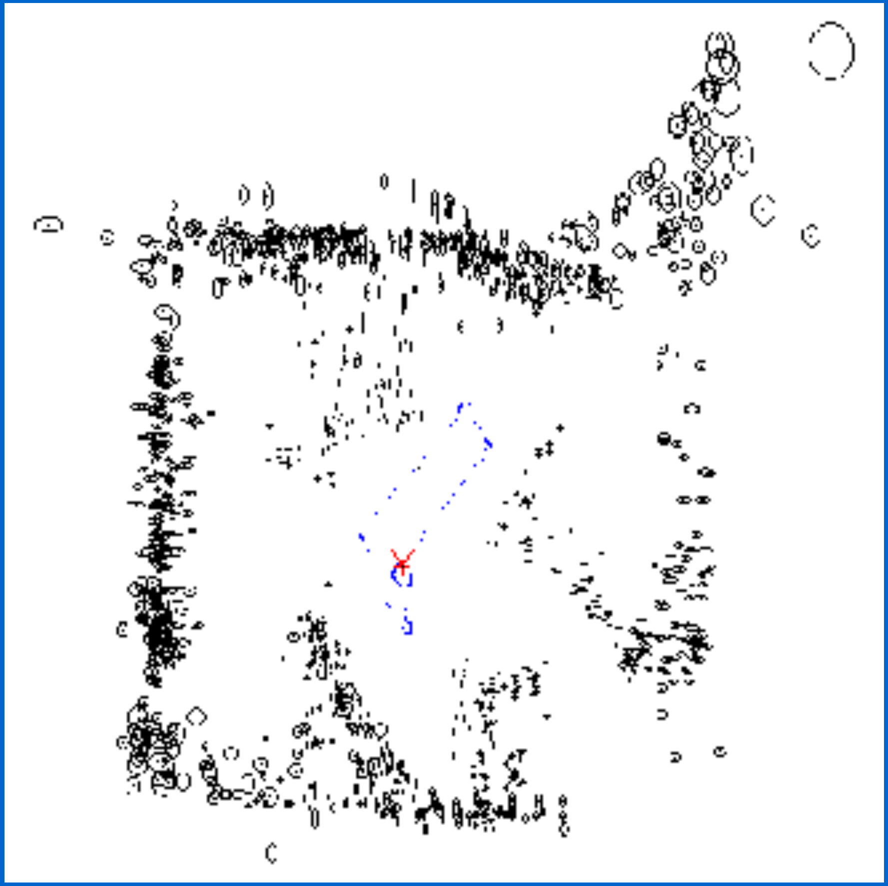
Object Categorization



Location recognition

Robot Localization

Map continuously built over time



SIFT Computation – Steps

(1) Scale-space extrema detection

- Extract scale and rotation invariant interest points (i.e., keypoints).

(2) Keypoint localization

- Determine location and scale for each interest point.
- Eliminate "weak" keypoints

(3) Orientation assignment

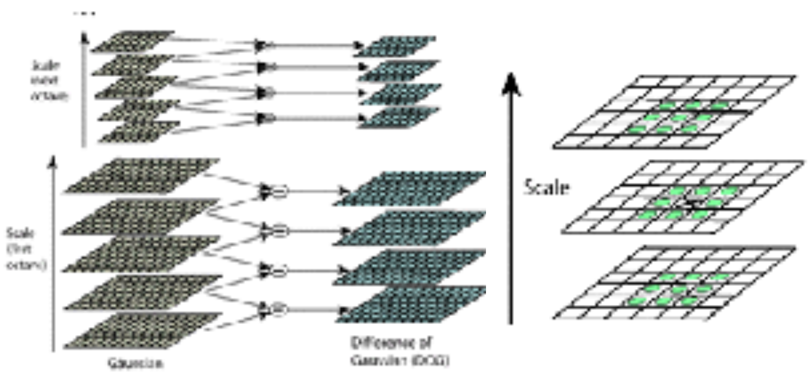
- Assign one or more orientations to each keypoint.

(4) Keypoint descriptor

- Use local image gradients at the selected scale.

D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints", **International Journal of Computer Vision**, 60(2):91-110, 2004.

1. Feature detection - Key point detection



Scale of an Image: $L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$

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

Difference-of-Gaussian function:

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

$k = \sqrt{2}$

1. Key point localization

- Detailed fit using data surrounding the keypoint to Localize extrema by fitting a quadratic, to nearby data for location, scale and ratio of principal curvatures

1) Sub-pixel/sub-scale interpolation using Taylor expansion

$$D(\overrightarrow{x}) = D + \frac{\partial D^T}{\partial \overrightarrow{x}}\overrightarrow{x} + \frac{1}{2}\overrightarrow{x}^T \frac{\partial^2 D}{\partial \overrightarrow{x}^2}\overrightarrow{x} \quad ; \quad \overrightarrow{x} = (x, y, \sigma)^T$$

Location of the extrema, $\hat{x} = -\frac{\partial^2 D}{\partial x^2}^{-1} \frac{\partial D}{\partial x}$

by taking a derivative and setting it to zero

1. Key point localization

1) Sub-pixel/sub-scale interpolation using Taylor expansion

$$D(\overrightarrow{x}) = D + \frac{\partial D^T}{\partial \overrightarrow{x}} \overrightarrow{x} + \frac{1}{2} \overrightarrow{x}^T \frac{\partial^2 D}{\partial \overrightarrow{x}^2} \overrightarrow{x} \qquad ; \qquad \overrightarrow{x} = (x, y, \sigma)^T$$

Location of the extrema, $\hat{x} = - \frac{\partial^2 D}{\partial x^2}^{-1} \frac{\partial D}{\partial x}$

$$\frac{\partial D}{\partial x} = \begin{bmatrix} \frac{\partial D}{\partial x} \\ \frac{\partial D}{\partial y} \\ \frac{\partial D}{\partial \sigma} \end{bmatrix} = \begin{bmatrix} \frac{D(x+1,y,\sigma) - D(x-1,y,\sigma)}{2} \\ \frac{D(x,y+1,\sigma) - D(x,y-1,\sigma)}{2} \\ \frac{D(x,y,\sigma+1) - D(x,y,\sigma-1)}{2} \end{bmatrix}$$

$$D(\hat{x}) = D + \frac{1}{2} \frac{\partial D}{\partial x} \hat{x}$$

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

key points with low contrast

1. Key point localization - Eliminating edge response

1) Principal curvatures can be computed from a 2 x 2 Hessian matrix

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

$$\frac{\partial D}{\partial x} = \begin{bmatrix} \frac{\partial D}{\partial x} \\ \frac{\partial D}{\partial y} \\ \frac{\partial D}{\partial \sigma} \end{bmatrix} = \begin{bmatrix} \frac{D(x+1, y, \sigma) - D(x-1, y, \sigma)}{2} \\ \frac{D(x, y+1, \sigma) - D(x, y-1, \sigma)}{2} \\ \frac{D(x, y, \sigma+1) - D(x, y, \sigma-1)}{2} \end{bmatrix}$$

1. Key point localization - Eliminating edge response

1) Principal curvatures can be computed from a 2 x 2 Hessian matrix

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

$$\frac{\partial D}{\partial x} = \begin{bmatrix} \frac{\partial D}{\partial x} \\ \frac{\partial D}{\partial y} \\ \frac{\partial D}{\partial \sigma} \end{bmatrix} = \begin{bmatrix} \frac{D(x+1,y,\sigma) - D(x-1,y,\sigma)}{2} \\ \frac{D(x,y+1,\sigma) - D(x,y-1,\sigma)}{2} \\ \frac{D(x,y,\sigma+1) - D(x,y,\sigma-1)}{2} \end{bmatrix}$$

$$D_{xy} = \frac{\frac{D(x+1,y+1,\sigma) - D(x-1,y+1,\sigma)}{2} - \frac{D(x+1,y-1,\sigma) - D(x-1,y-1,\sigma)}{2}}{2}$$

1. Feature detection - Keypoint localization

- Discard low-contrast/edge points
  1) Low contrast: discard keypoints with threshold < 0.03
  2) Edge points: high contrast in one direction, low in the other → compute principal curvatures from eigenvalues of 2x2 Hessian matrix, and limit ratio

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

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

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

$$r = \frac{\alpha}{\beta} \qquad \frac{\mathrm{Tr}(\mathbf{H})^2}{\mathrm{Det}(\mathbf{H})} < \frac{(r+1)^2}{r} \qquad r = 10$$

2. Orientation Assignment

- Assign orientation to keypoints
  - Assign canonical orientation at peak of smoothed histogram

Gradient magnitude,

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

Orientation,

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

2. Orientation Assignment

- Assign orientation to keypoints
  - Create histogram of local gradient directions computed at selected scale

Orientation histogram has 36 bins each covering 10 degrees

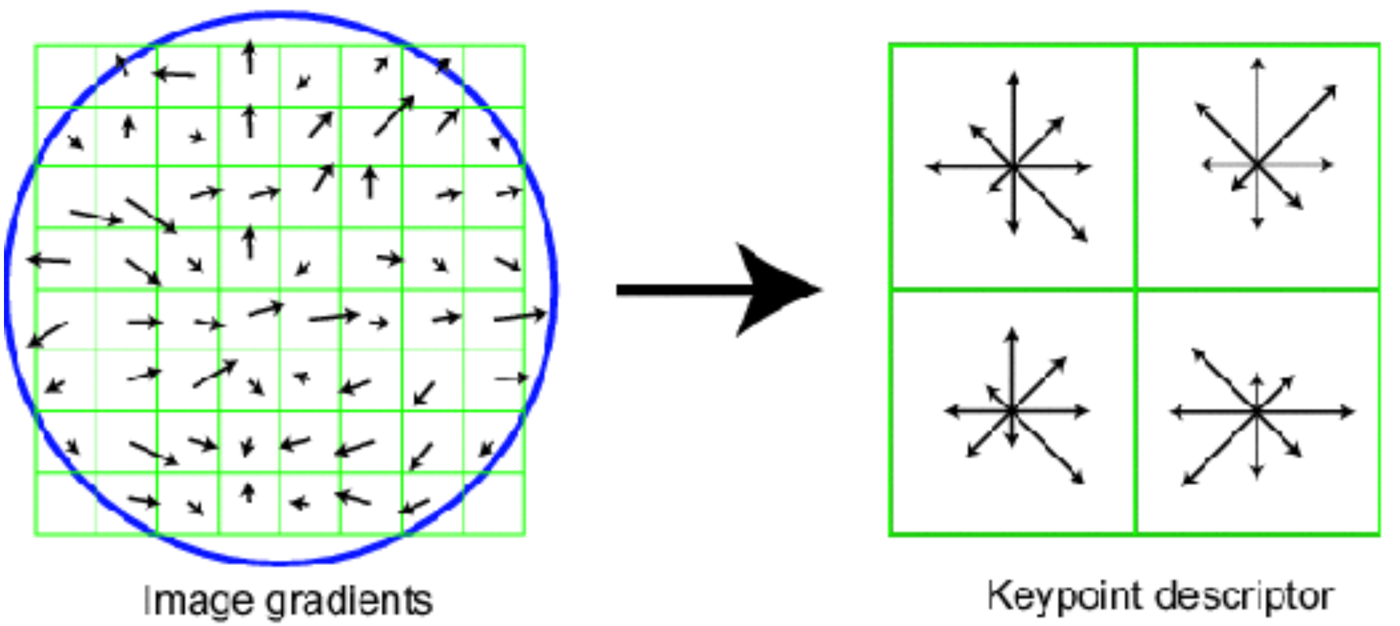
Peaks in the orientation histogram correspond to dominant directions of local gradients.

Any other local peak, within 80% of the highest peak is also used to create a key point with that orientation.

There may be multiple key points with same location and scale but different orientation.

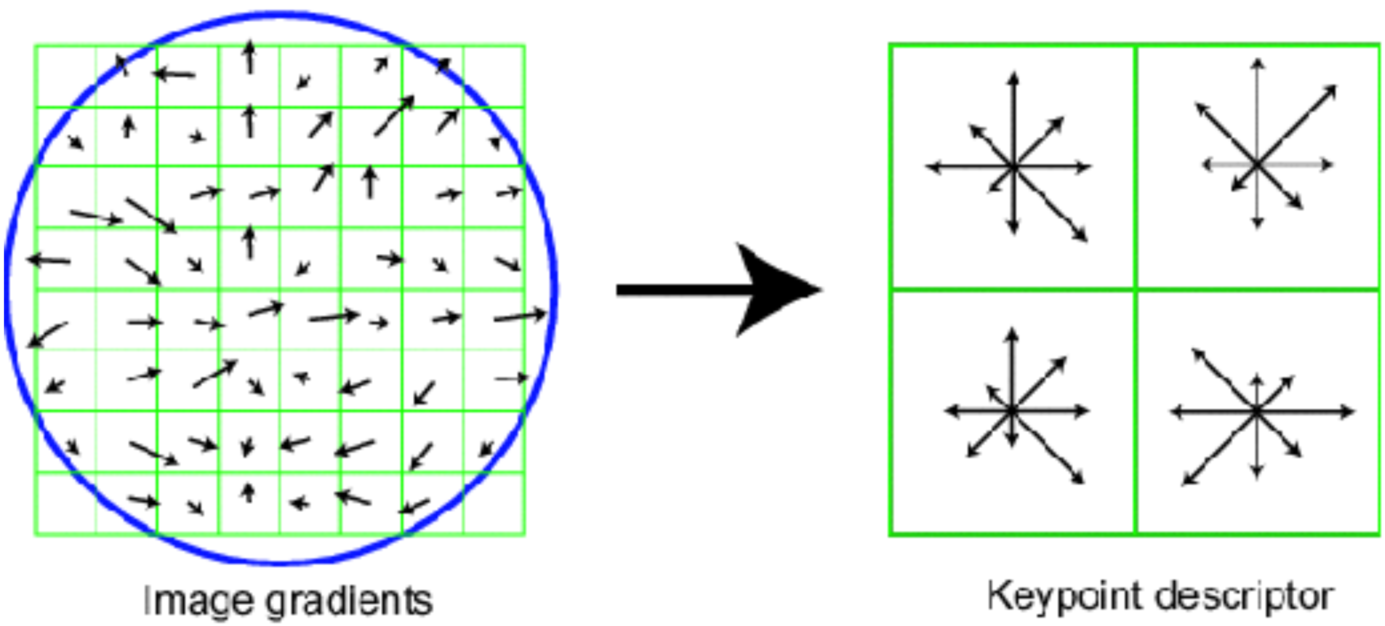
2. Feature description

- Construct SIFT descriptor
  - Create array of orientation histograms
  - 8 orientations x 4x4 histogram array = 128 dimensions



2. Feature description

- Advantage over simple correlation
  - less sensitive to illumination change
  - robust to deformation, viewpoint change

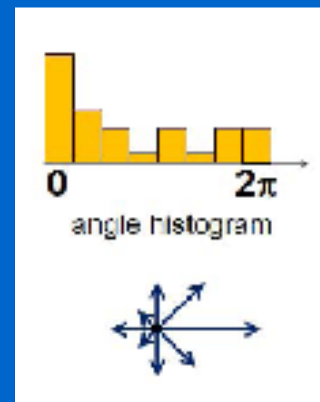


4. Keypoint Descriptor (cont’d)

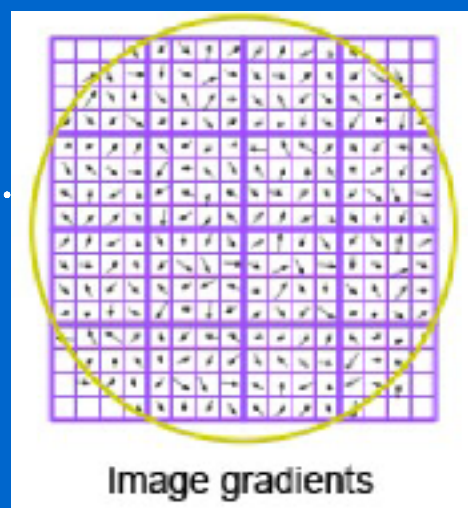
1. Take a 16 x16 window around detected interest point.

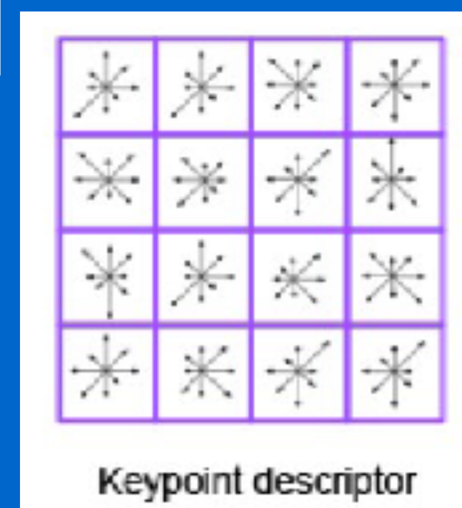
2. Divide into a 4x4 grid of cells.

3. Compute histogram in each cell.



(8 bins)

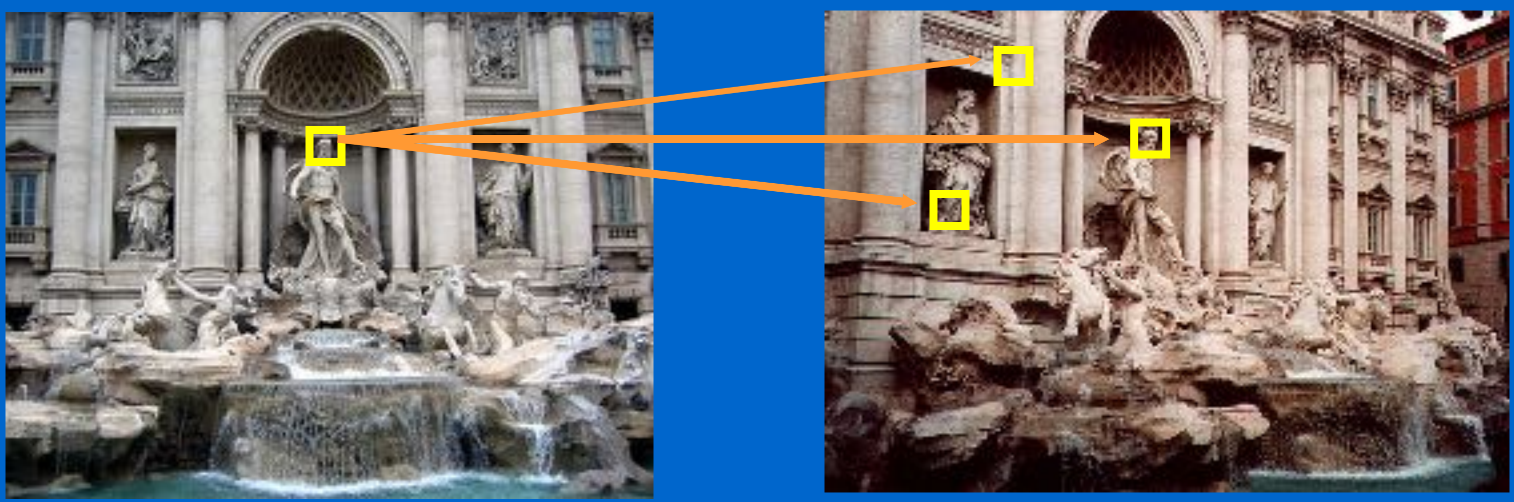




16 histograms x 8 orientations
= 128 features

Matching SIFT features

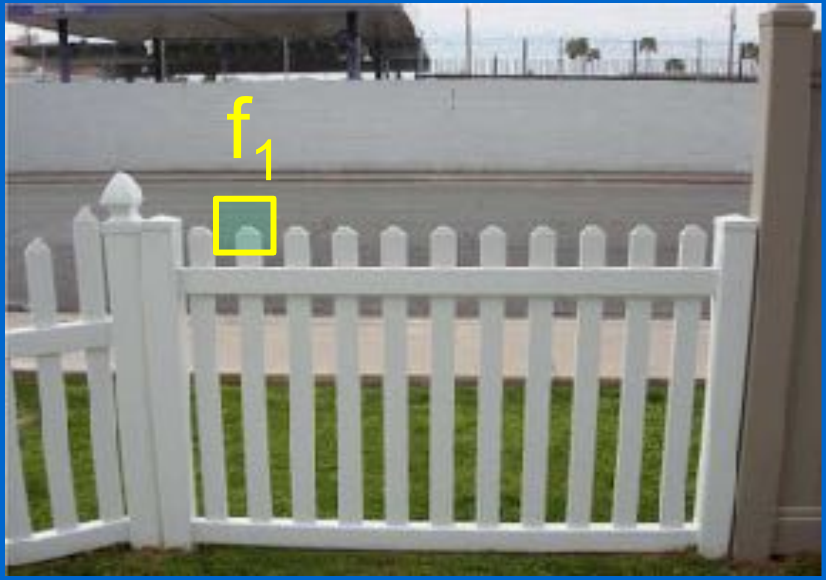
- Given a feature in I_1, how to find the best match in I_2?
  1. Define distance function that compares two descriptors.
  2. Test all the features in I_2, find the one with min distance.



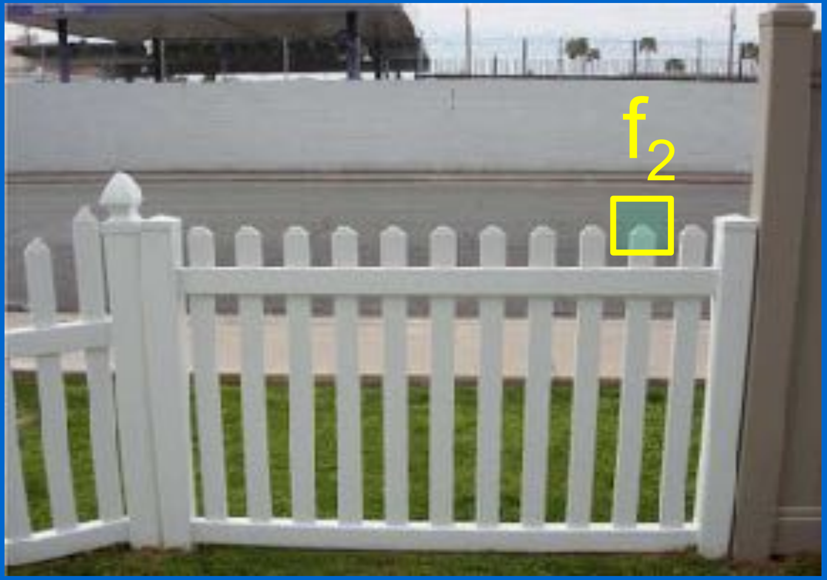
I_1 I_2

Matching SIFT features (cont'd)

- What distance function should we use?
  - Use SDD(f_1, f_2)=$\sum_{i=0}^{N-1}(f_{1i} - f_{2i})^2$ (i.e., sum of squared differences)
  - Can give good scores to very ambiguous (bad) matches



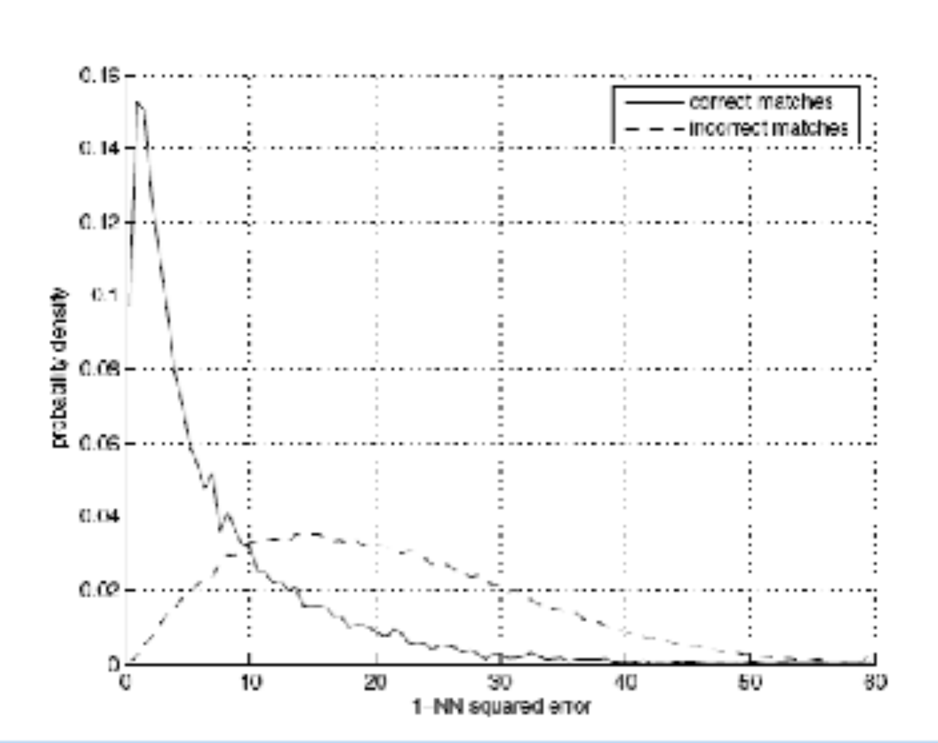
I_1



I_2

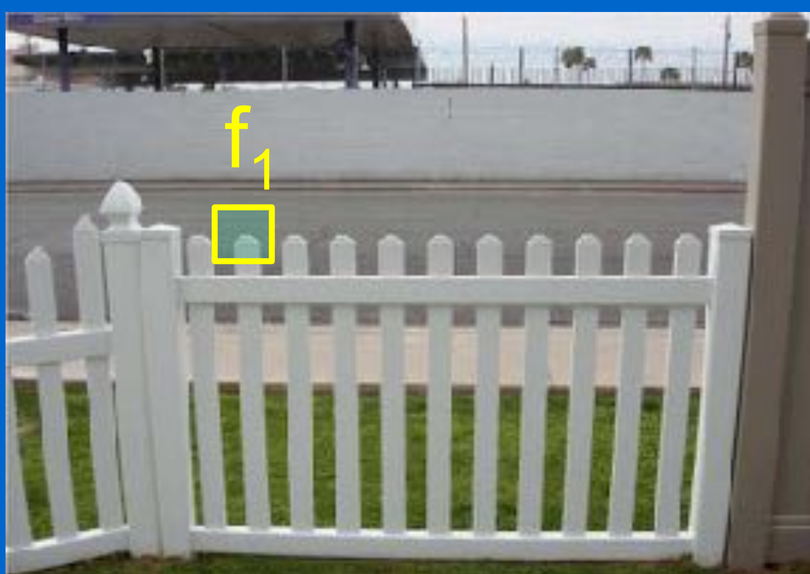
Matching SIFT features (cont'd)

- Accept a match if SSD(f1,f2) < t
- How do we choose t?

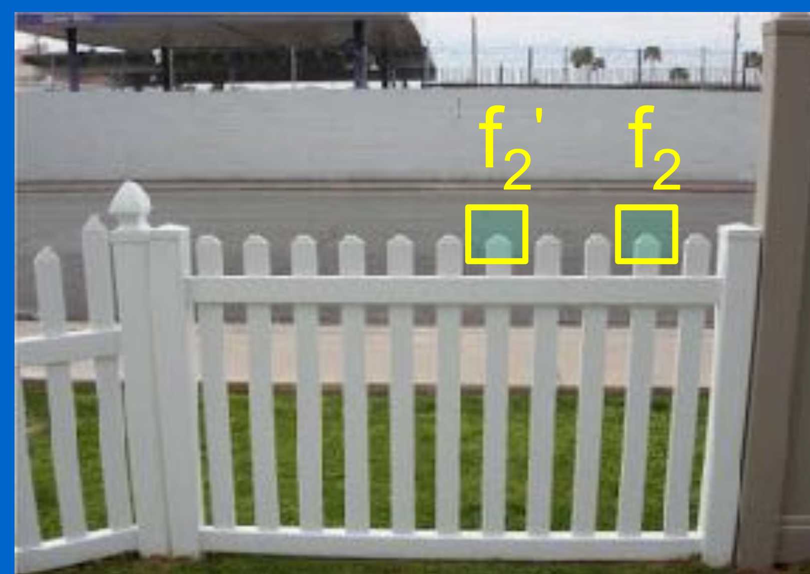


Matching SIFT features (cont'd)

- A better distance measure is the following:
  - $SSD(f_1, f_2) / SSD(f_1, f_2')$
    - f_2 is best SSD match to f_1 in I_2
    - f_2' is 2nd best SSD match to f_1 in I_2



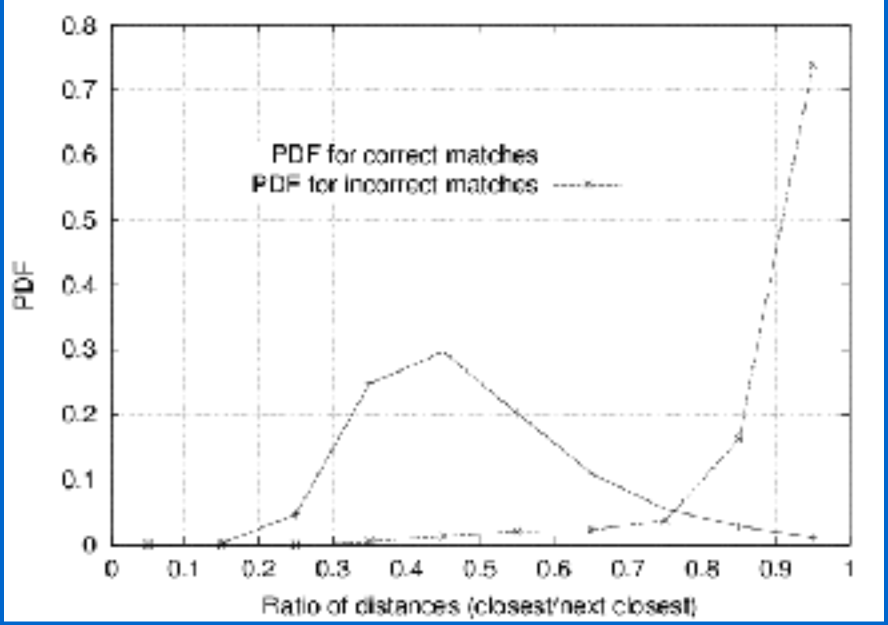
I_1



I_2

Matching SIFT features (cont'd)

- Accept a match if $SSD(f_1, f_2) / SSD(f_1, f_2') < t$
- t=0.8 has given good results in object recognition.
  - 90% of false matches were eliminated.
  - Less than 5% of correct matches were discarded



Variations of SIFT features

- PCA-SIFT
- SURF
- GLOH

SURF: Speeded Up Robust Features

- Speed-up computations by fast approximation of (i) Hessian matrix and (ii) descriptor using "integral images".

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

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

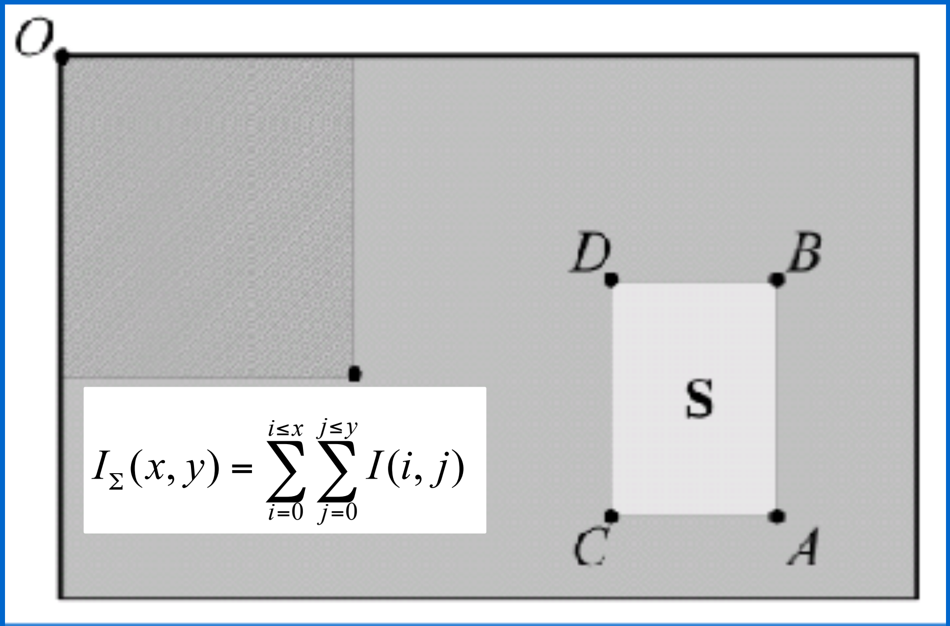
SIFT

- What is an "integral image"?

Herbert Bay, Tinne Tuytelaars, and Luc Van Gool, "SURF: Speeded Up Robust Features", **European Computer Vision Conference (ECCV),** 2006.

Integral Image

- The integral image $I_{\Sigma}(x,y)$ of an image I(x, y) represents the sum of all pixels in I(x,y) of a rectangular region formed by (0,0) and (x,y).



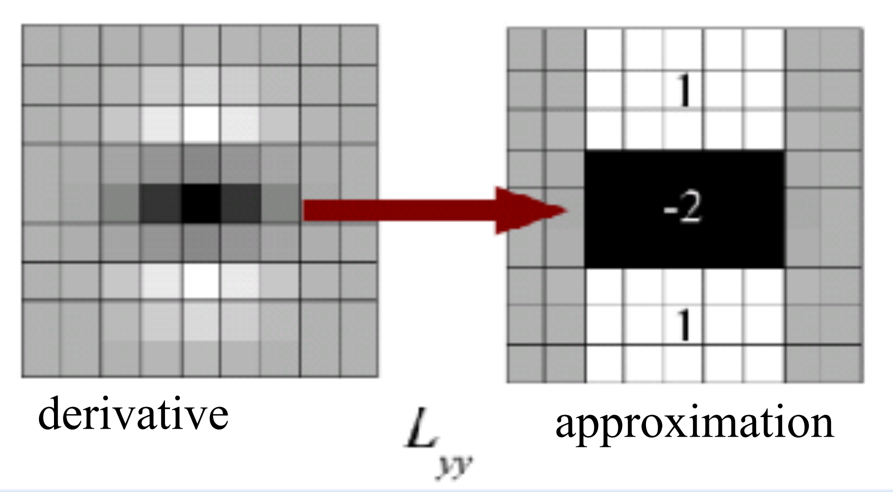
Using integral images, it takes only four array references to calculate the sum of pixels over a rectangular region of any size.

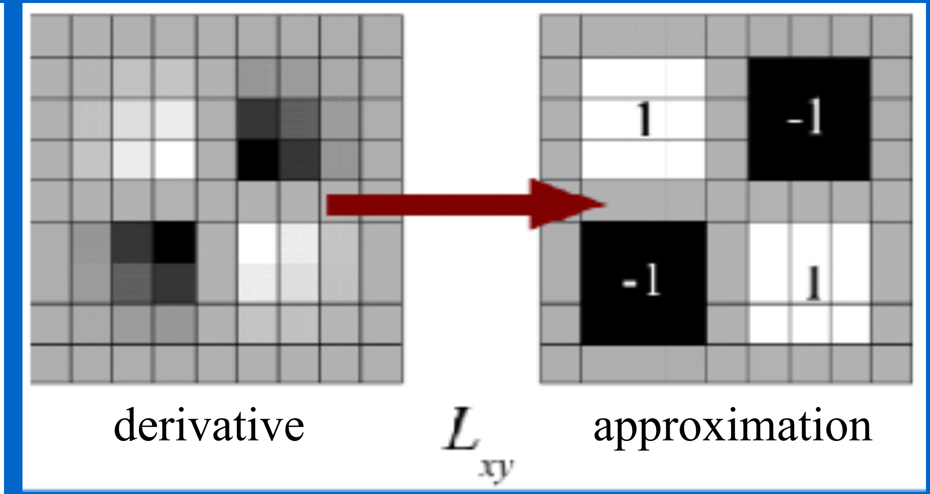
$$\mathbf{S} = A - B - C + D$$

SURF: Speeded Up Robust Features (cont’d)

- Approximate L_{xx}, L_{yy}, and L_{xy} using box filters.

(box filters shown are 9 x 9 – good approximations for a Gaussian with σ=1.2)

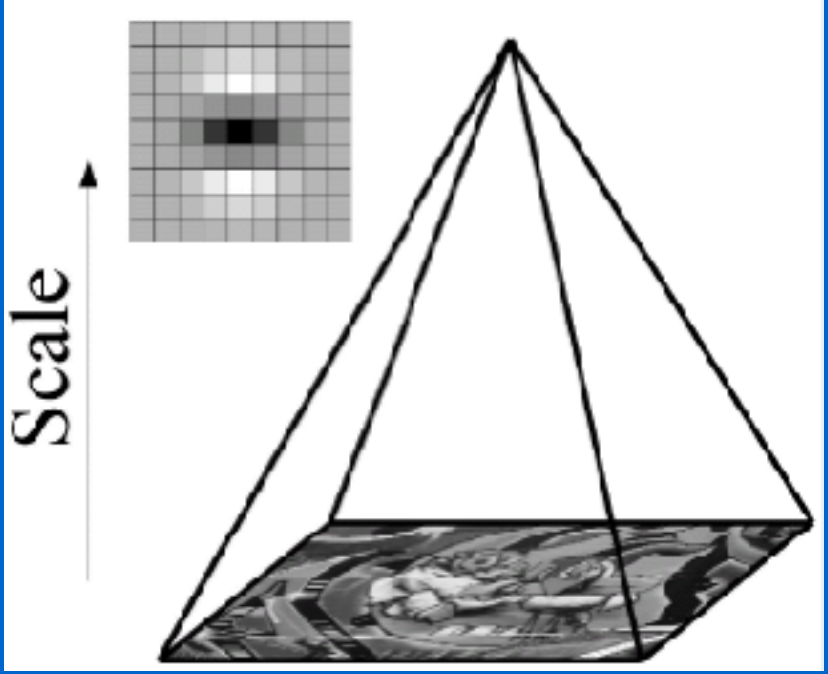




- Can be computed very fast using integral images!

SURF: Speeded Up Robust Features (cont'd)

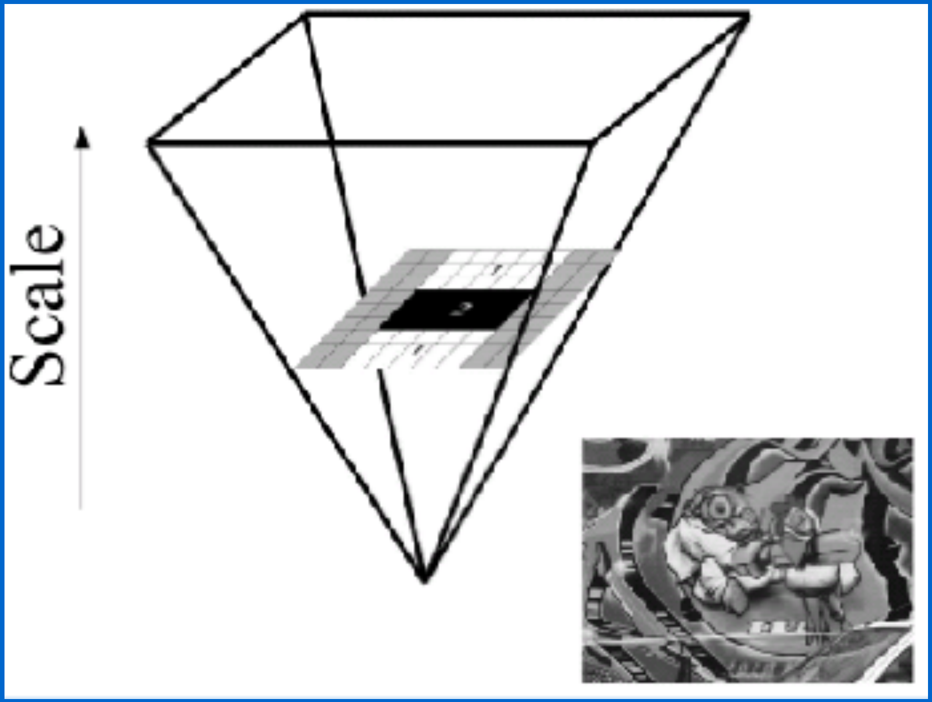
- In SIFT, images are repeatedly smoothed with a Gaussian and subsequently sub-sampled in order to achieve a higher level of the pyramid.



SURF: Speeded Up Robust Features (cont'd)

- Alternatively, we can use filters of larger size on the original image.

- Due to using integral images, filters of any size can be applied at exactly the same speed!



(see Tuytelaars' paper for details)

SURF: Speeded Up Robust Features (cont'd)

- Approximation of H:

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

Using DoG

$$SIFT:\ H_{approx}^{SIFT} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{yx} & D_{yy} \end{bmatrix}$$

Using box filters

$$SURF:\ H_{approx}^{SURF} = \begin{bmatrix} \hat{L}_{xx} & \hat{L}_{xy} \\ \hat{L}_{yx} & \hat{L}_{yy} \end{bmatrix}$$

SURF: Speeded Up Robust Features (cont'd)

- Instead of using a different measure for selecting the location and scale of interest points (e.g., Hessian and DOG in SIFT), SURF uses the determinant of H^{SURF}_{approx} to find both.

- Determinant elements must be weighted to obtain a good approximation:

$$\det(H^{SURF}_{approx}) = \hat{L}_{xx}\hat{L}_{yy} - (0.9\hat{L}_{xy})^2$$

Harris corners

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

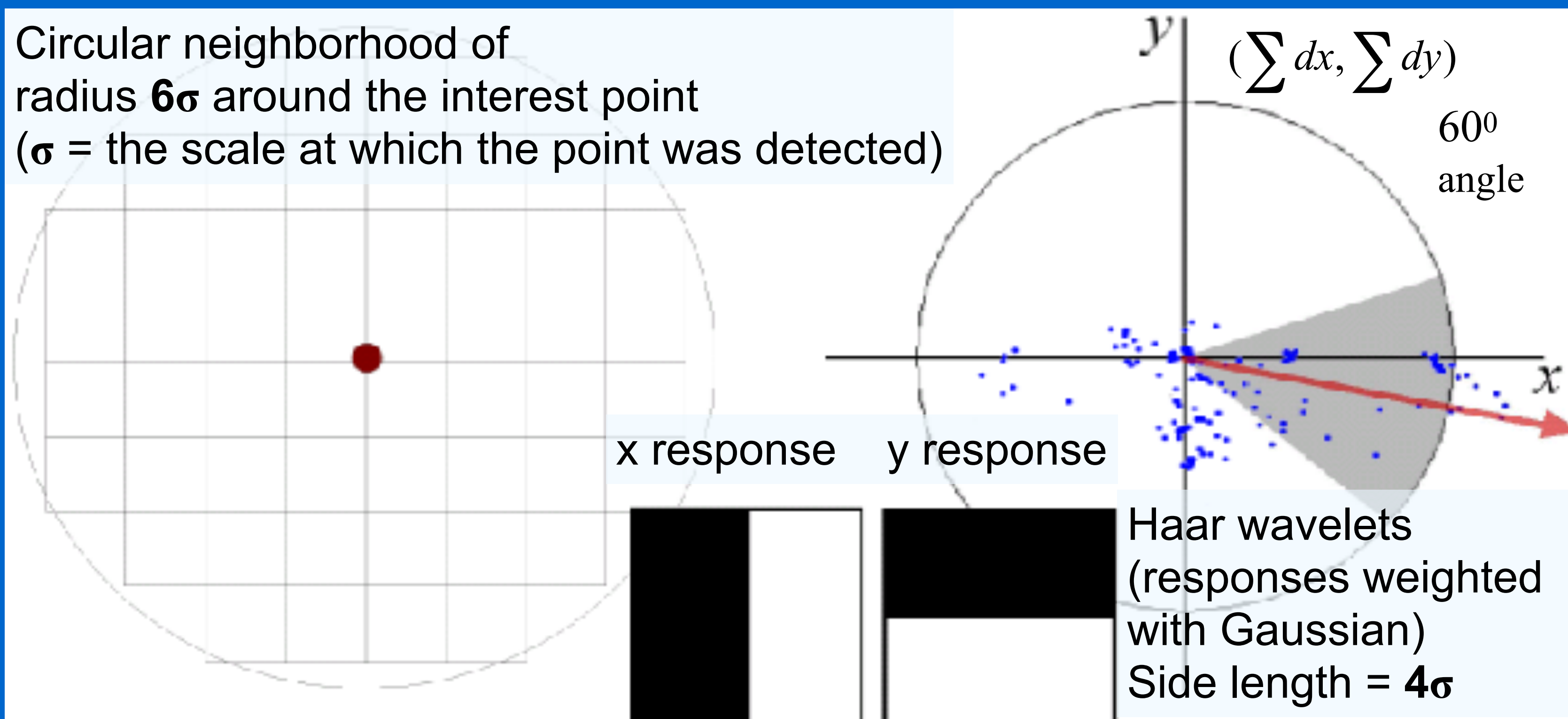
$$\mathrm{Det}(\mathbf{H}) = D_{xx}D_{yy} - (D_{xy})^2 = \alpha\beta.$$

SURF: Speeded Up Robust Features (cont'd)

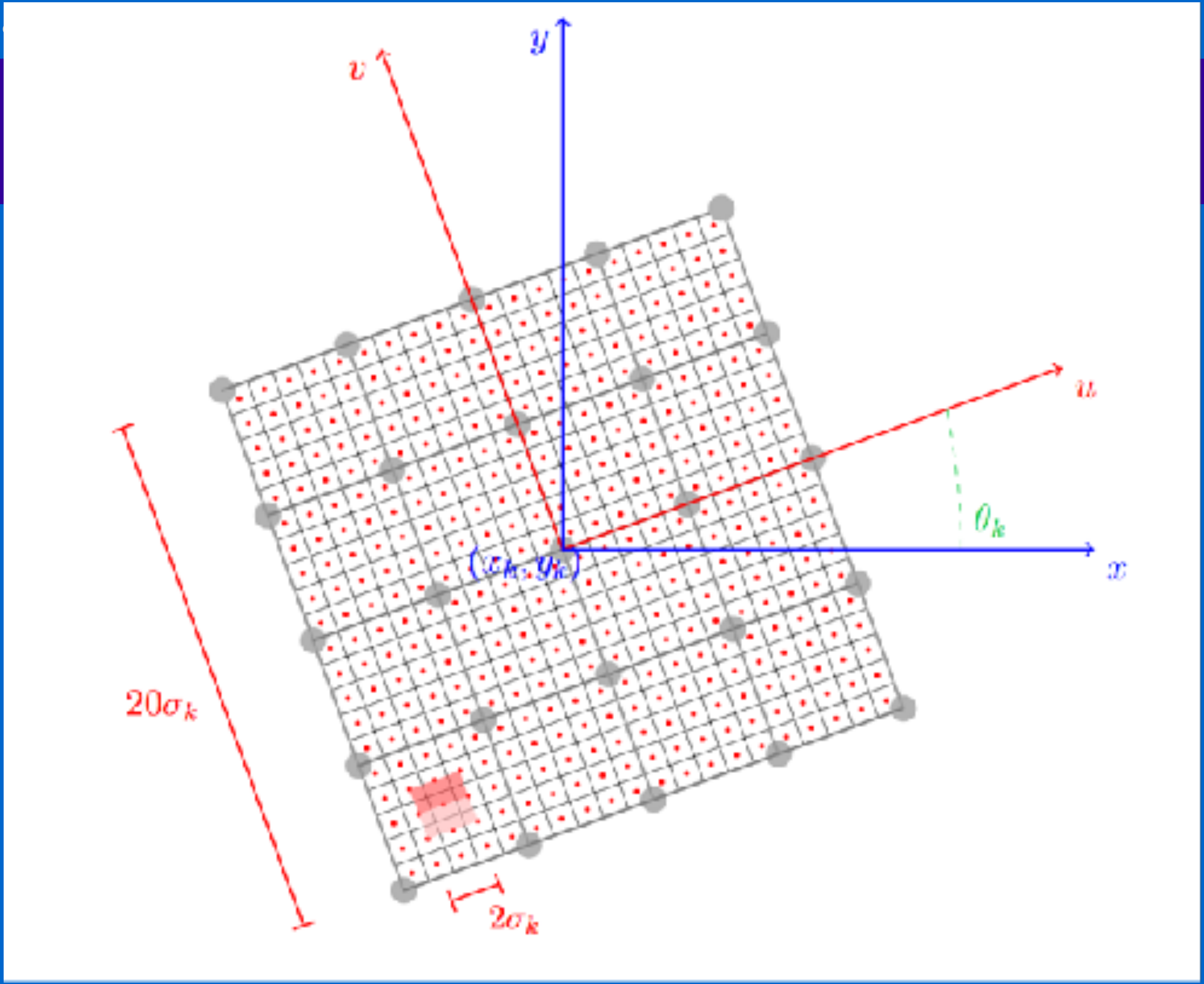
- Once interest points have been localized both in space and scale, the next steps are:
  - (1) Orientation assignment
  - (2) Keypoint descriptor

SURF: Speeded Up Robust Features (cont'd)

- Orientation assignment



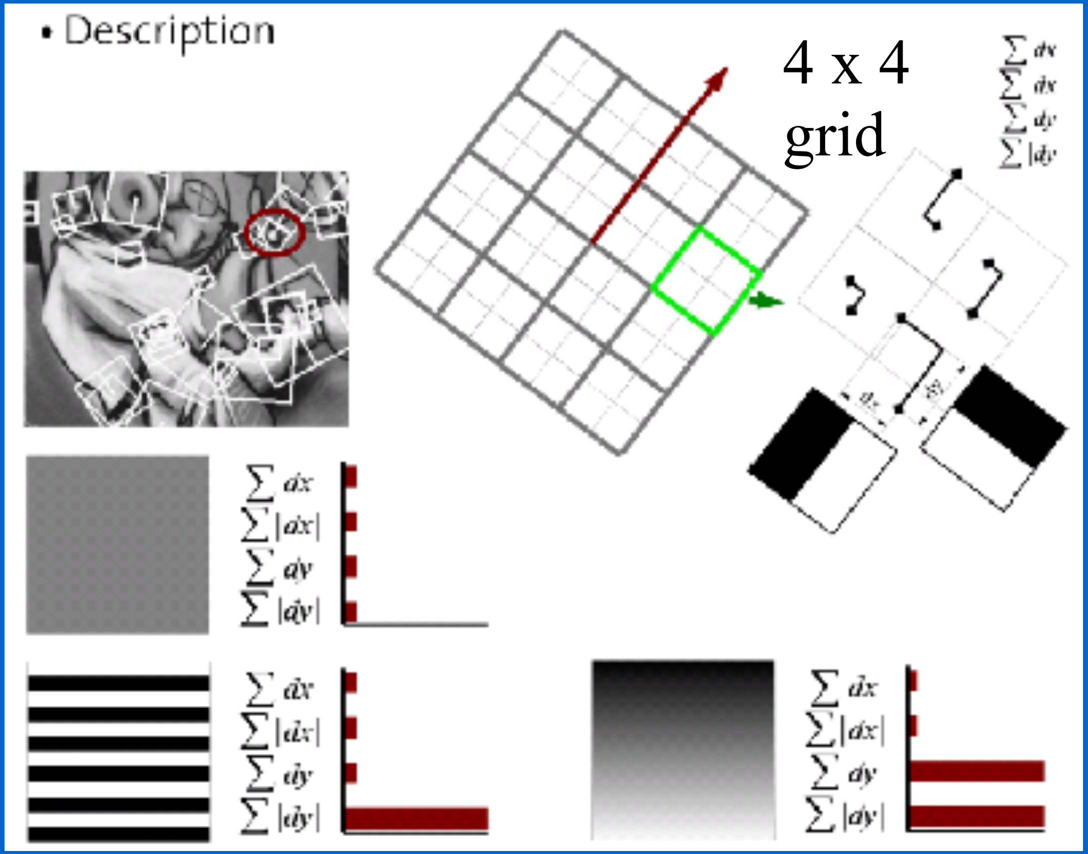
Can be computed very fast using integral images!



$$S_k : \begin{pmatrix} u \\ v \end{pmatrix} \mapsto \begin{pmatrix} x \\ y \end{pmatrix} = \sigma_k \cdot R_{\theta_k} \begin{pmatrix} u \\ v \end{pmatrix} + \begin{pmatrix} x_k \\ y_k \end{pmatrix} , \qquad \text{where } R_\alpha = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} .$$

SURF: Speeded Up Robust Features (cont'd)

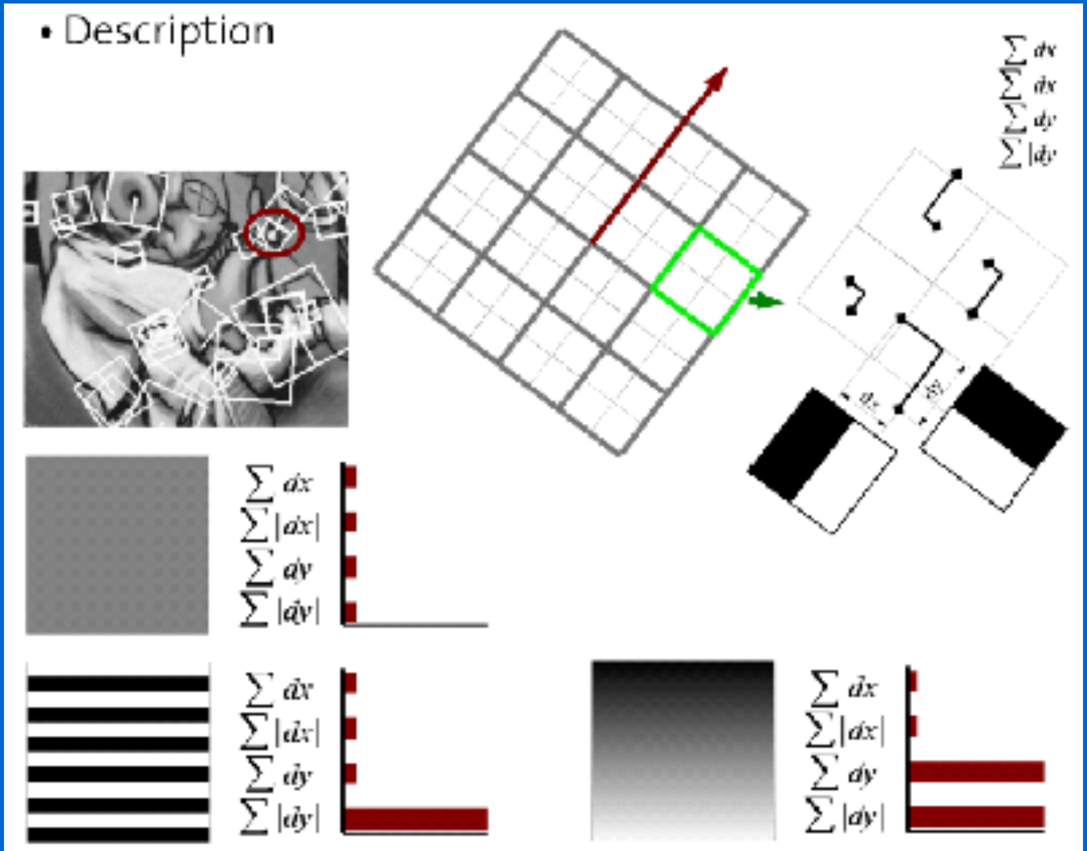
- Keypoint descriptor (square region of size 20σ)



- Sum the response over each sub-region for d_x and d_y separately.
- To bring in information about the polarity of the intensity changes, extract the sum of absolute value of the responses too.

Feature vector size:
4 x 16 = 64

SURF: Speeded Up Robust Features (cont'd)



- SURF-128
  - The sum of d_x and $|d_x|$ are computed separately for points where $d_y < 0$ and $d_y > 0$
  - Similarly for the sum of d_y and $|d_y|$
  - **More discriminatory!**

SURF: Speeded Up Robust Features

- Has been reported to be 3 times faster than SIFT.
- Less robust to illumination and viewpoint changes compared to SIFT.

K. Mikolajczyk and C. Schmid,"A Performance Evaluation of Local Descriptors", **IEEE Transactions on Pattern Analysis and Machine Intelligence**, vol. 27, no. 10, pp. 1615-1630, 2005.

OpenCV packages

- FindHomography
- FlannBasedMatcher FLANN stands for Fast Library for Approximate Nearest Neighbors. It contains a collection of algorithms optimized for fast nearest neighbor search in large datasets and for high dimensional features.
  - Uses k-d trees