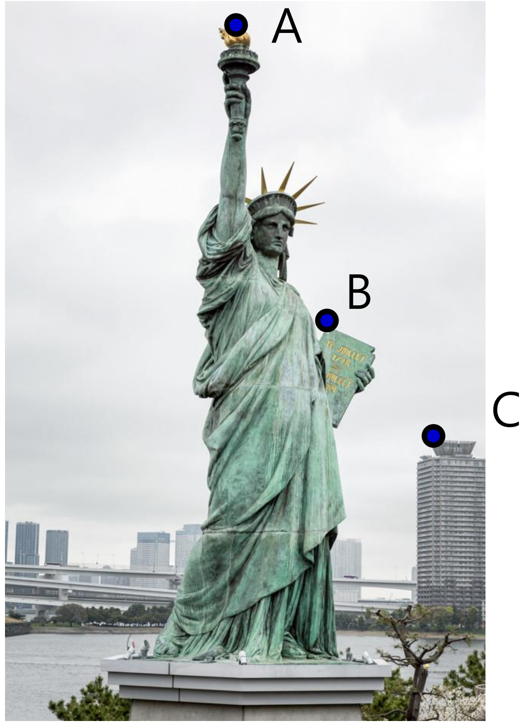
3D Data Processing

Feature descriptor

Hyoseok Hwang

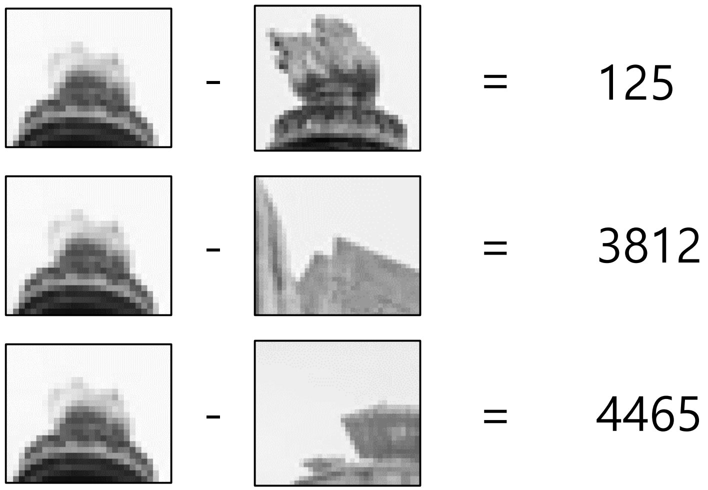
Feature matching

- How can we find a part of one image that matches another?
- How can we judge that two points are similar?
- How can we define the "similarity" of local features?

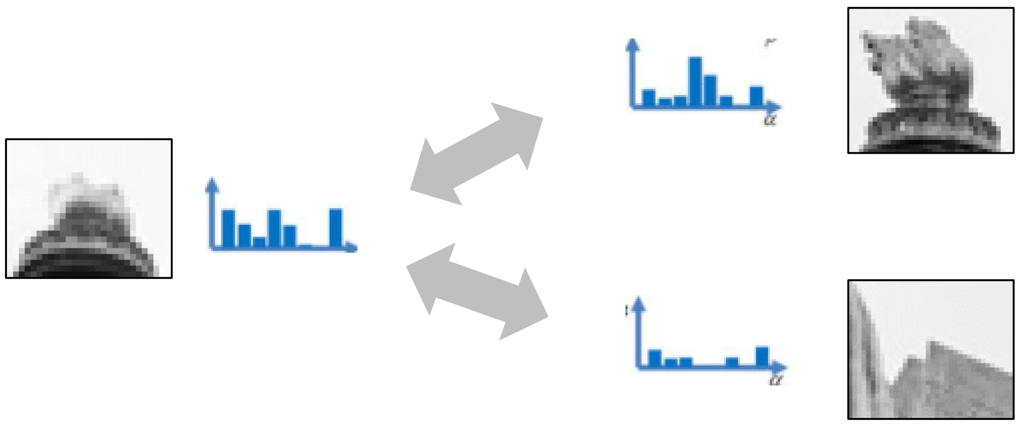


Feature matching

- We can measure similarity by
  - Template matching
    - Calculate pixel-wise differences of templates centered on the feature point.
  - Distance of descriptors
    - Calculate the similarity between descriptors describing feature points.



template matching



Descriptor matching

Template matching

- A template
  - 2D matrix centered on a point.

- The matching process involves computation of the similarity measure for each disparity value, followed by an aggregation within the square window.

Template matching

- Sum of Absolute Differences (SAD)

$$SAD(i,j) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \left| I(i+m, j+n) - T(m,n) \right|$$

- Sum of Squared Differences (SSD)

$$SSD(i,j) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \left(I(i+m, j+n) - T(m,n) \right)^2$$

- Normalized Cross Correlation (NCC)

$$NCC(i,j) = \frac{\sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I(i+m, j+n) \cdot T(m,n)}{\left(\sqrt{\sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I(i+m, j+n)^2}\right) \cdot \left(\sqrt{\sum_{m=0}^{M-1} \sum_{n=0}^{N-1} T(m,n)^2}\right)}$$

Template matching

- An example of template matching (SSD)

$$\begin{matrix} 7 & 9 & 8 \\ 5 & 4 & 6 \\ 9 & 8 & 2 \end{matrix} \quad \text{versus} \quad \begin{matrix} 8 & 7 & 9 \\ 7 & 5 & 4 \\ 7 & 5 & 4 \end{matrix} \quad \longrightarrow \quad \begin{aligned} \text{SSD} = {} & (7-8)^2 + (9-7)^2 + (8-9)^2 + \\ & (5-7)^2 + (4-5)^2 + (6-4)^2 + \\ & (9-7)^2 + (8-5)^2 + (2-4)^2 \\ = {} & 1 + 4 + 1 + 4 + 1 + 4 + 4 + 9 + 4 \\ = {} & 32 \end{aligned}$$

$$\begin{matrix} 7 & 9 & 8 \\ 5 & 4 & 6 \\ 9 & 8 & 2 \end{matrix} \quad \text{versus} \quad \begin{matrix} 8 & 7 & 10 \\ 6 & 5 & 4 \\ 10 & 7 & 1 \end{matrix} \quad \longrightarrow \quad \text{SSD} = 18$$

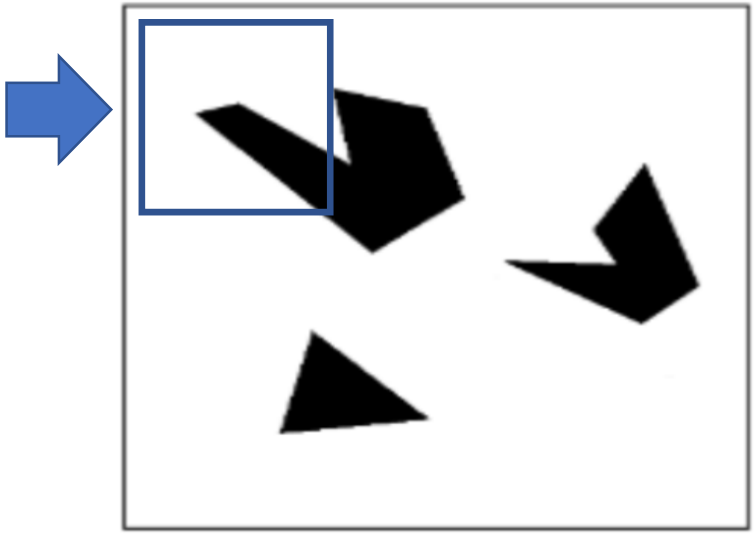
min SSD = 18 =>

take match windows:

$$\begin{matrix} 7 & 9 & 8 \\ 5 & 4 & 6 \\ 9 & 8 & 2 \end{matrix} \quad \text{and} \quad \begin{matrix} 8 & 7 & 10 \\ 6 & 5 & 4 \\ 10 & 7 & 1 \end{matrix}$$

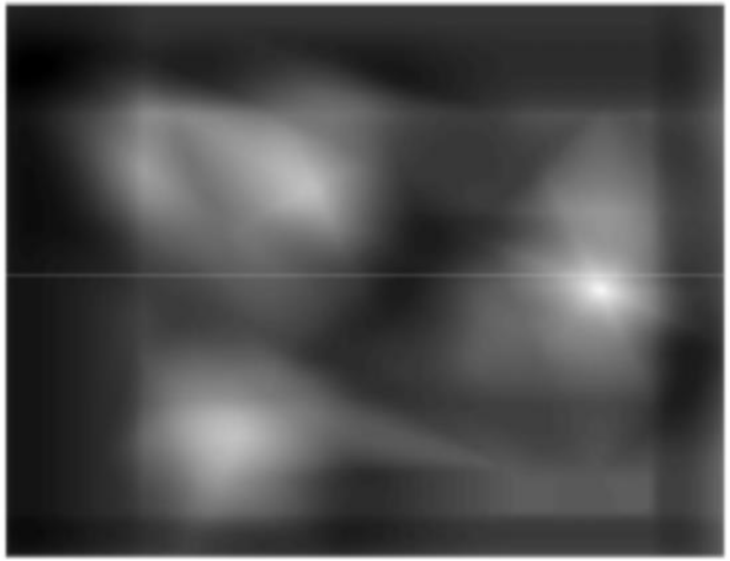
Template matching

- An example of template matching
  - Note, this result shows all results of NCC by sliding window, not among local features.
  - The maximum value of NCC is 1.



scene

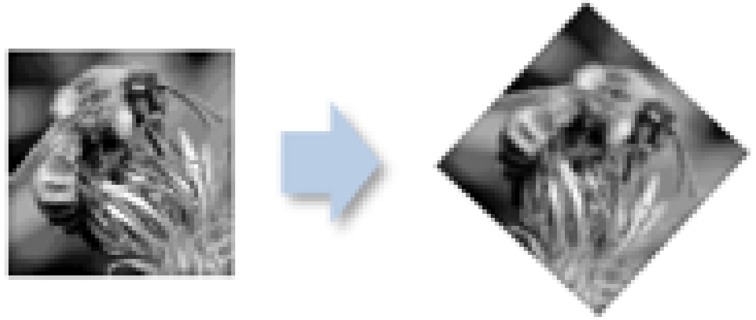
template



result

Template matching

- Disadvantages of template matching
  - Even if the points were extracted from the same position of the same object, the matching score is degraded if there are any of the following relationships.
    - Rotation
    - Scaling
    - Intensity change
      (NCC is invariant)
    - Affine transform



Feature Descriptor

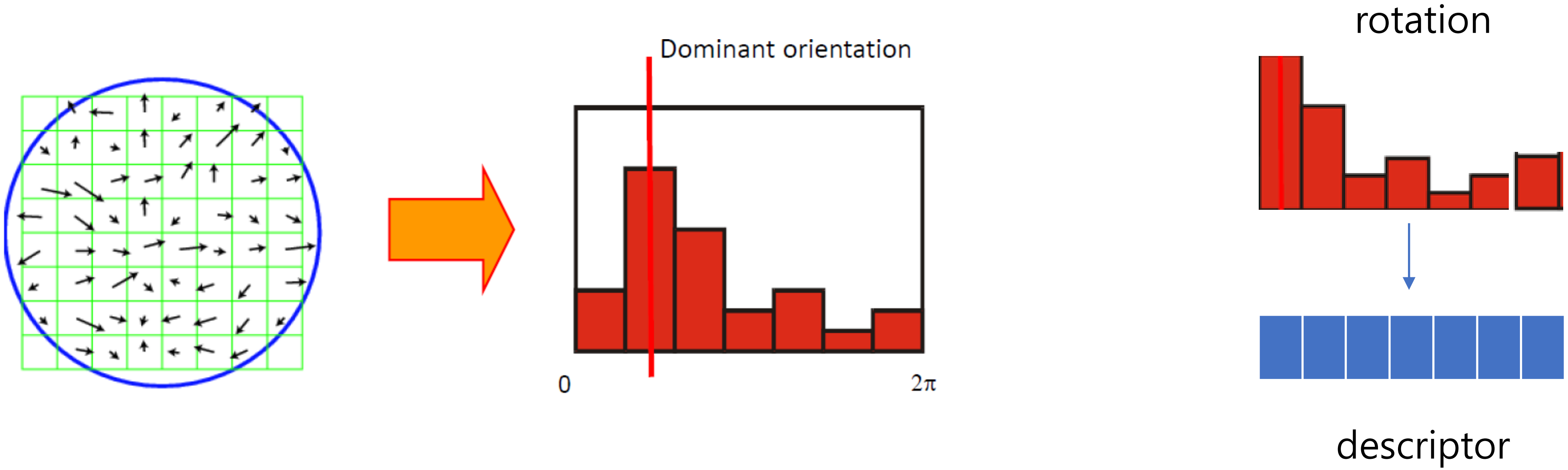
- Descriptor
  - Description of a feature
  - Usually expressed as a vector

- We can also regard a template as a kind of descriptors.
  - A template(2D matrix) can be modified to a vector.

- We need a better way to describe features which is robust image transform, i.e., rotation, scaling.

HOG Descriptor (Histogram of Oriented Gradient)

- Multiply the patch by a Gaussian kernel to make the shape circular
- Compute gradient vectors of each pixel
- Build histogram of gradient orientation → weighted by gradient magnitudes in constant angle units (hog descriptor)
- Extract all local maxima of HOG, then rotation



SIFT

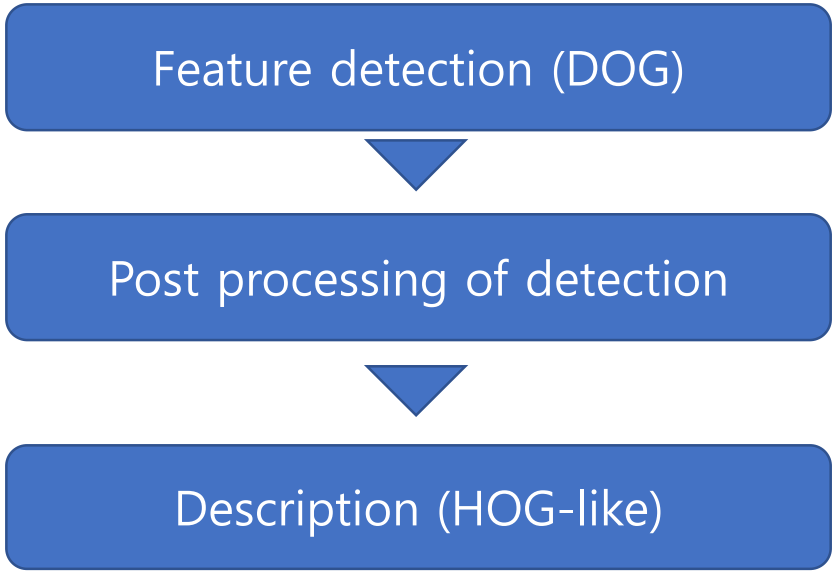
- Scale Invariant Feature Transform
  - By David Lowe (UBC)
  - Stands for scale invariant feature transform
  - Patented by university of British Columbia
    - Expired in March of 2020.
  - Similar to the one used in primate visual system (human, ape, monkey, etc.)
  - Transforms image data into scale-invariant coordinates

- Goal
  - Extracting(Detection & Description) distinctive invariant features
  - Invariance to image scale and rotation

D. Lowe. *Distinctive image features from scale-invariant key points.*, International Journal of Computer Vision 2004.

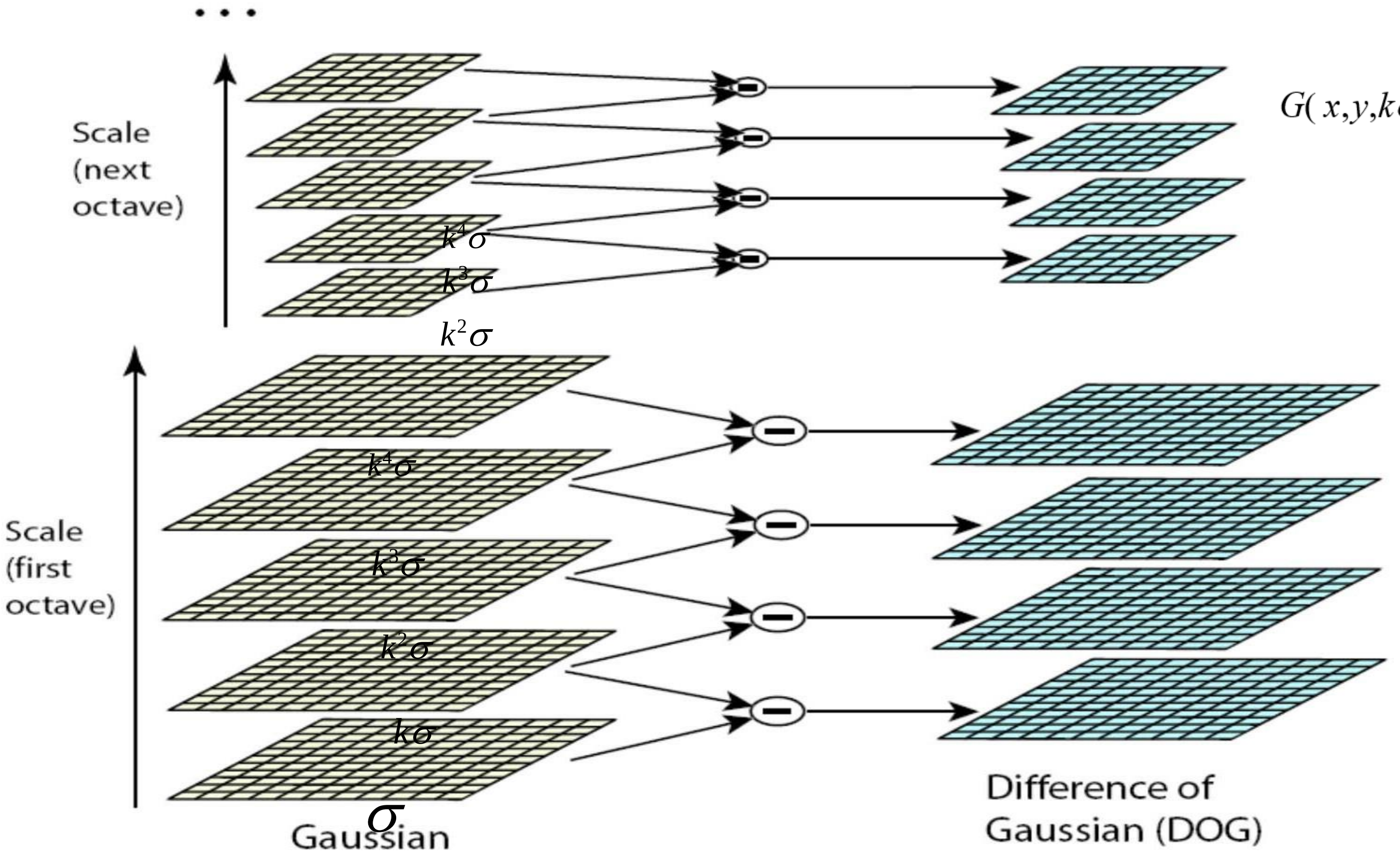
SIFT

- Process
  - Feature extraction
    - Extract candidate
    - Remove outliers
  - Description
    - Set major direction
      - Rotate image patch
    - Build 128 dimensional vectors with regional gradient



SIFT – Feature Detection

- Difference of Gaussian of Image Pyramid

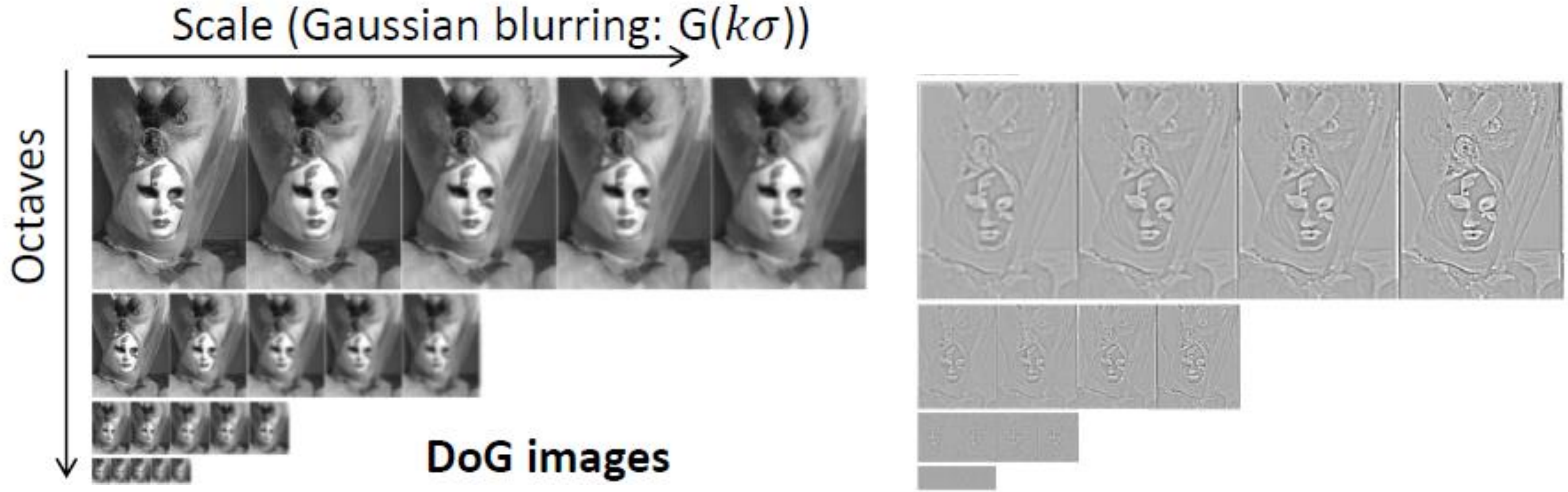


$$G(x,y,k\sigma) = \frac{1}{2\pi(k\sigma)^2} e^{-(x^2+y^2)/2k^2\sigma^2}$$

$$k = \sqrt{2}$$

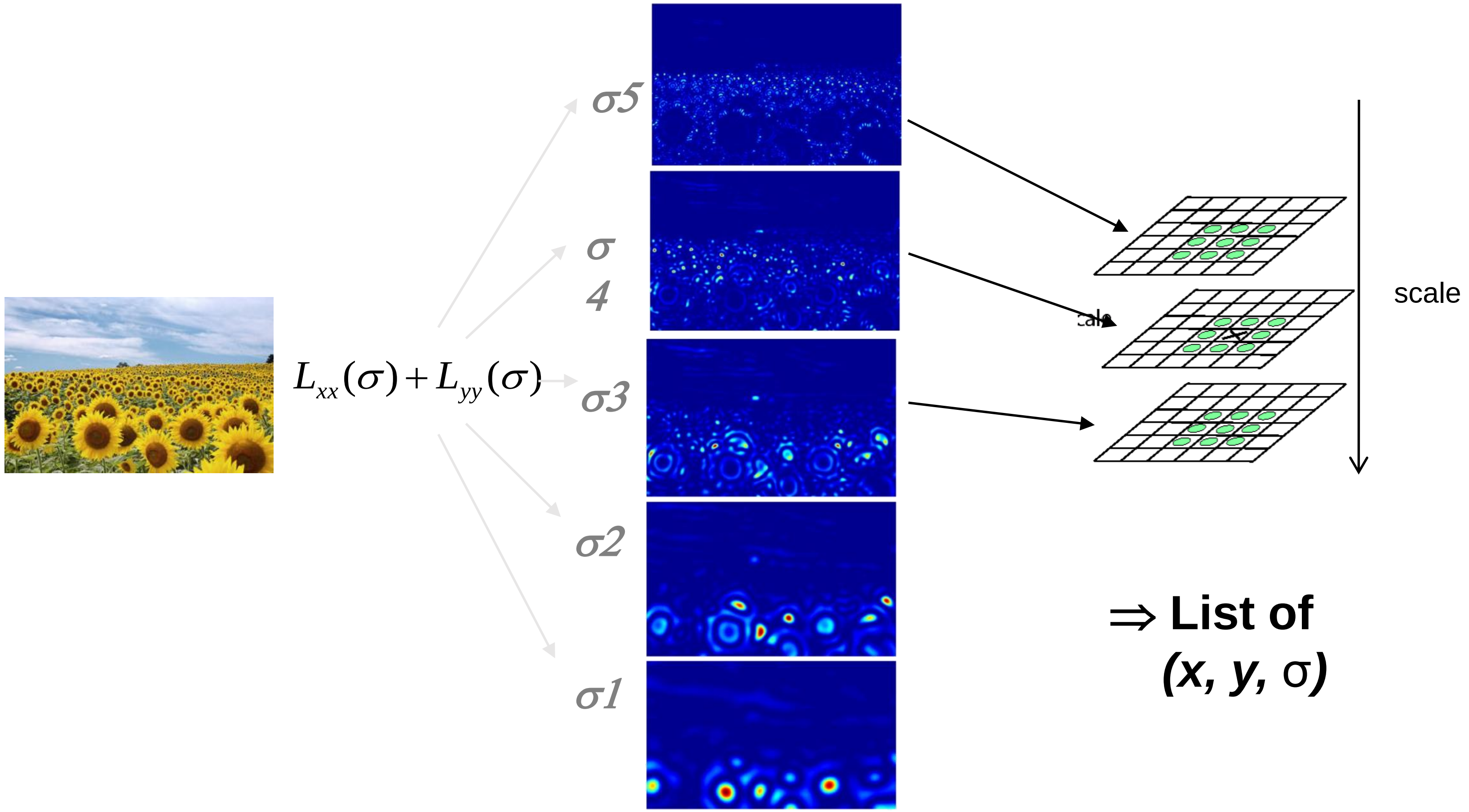
SIFT – Feature Detection

- Difference of Gaussian of Image Pyramid



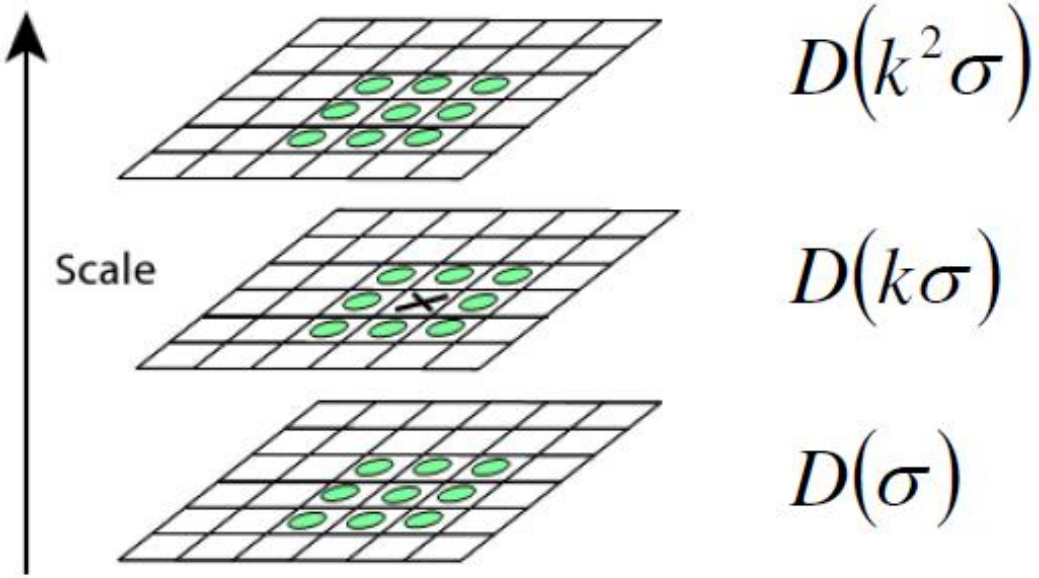
SIFT – Feature Detection

- Key point localization example



SIFT – Feature extraction

- Key point localization
  - Find all Extrema, that is minimum or maximum in 3x3x3 neighborhood
  - Therefore, the feature candidates are located except for the first and last DOG images.
  - Using a method of extracting a large number of candidates first, then removing outliers.



SIFT – Feature extraction

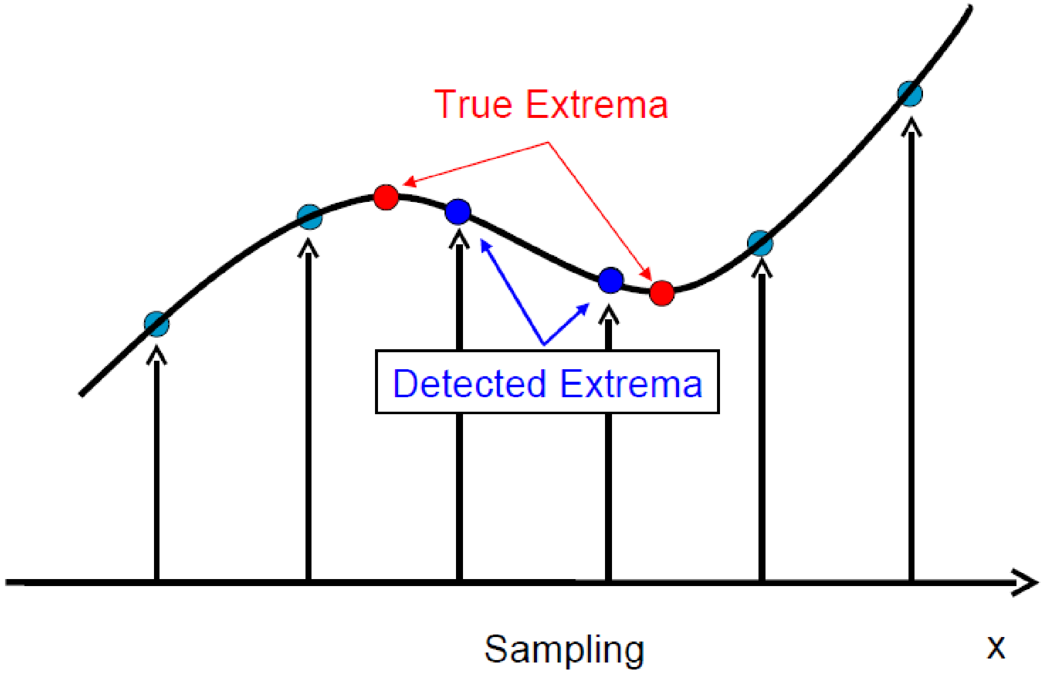
- Outlier removal #1 – Set accurate position
  - Sub Pixel Locate Potential Feature Points
  - Sub-pixel/sub-scale interpolation using Taylor expansion

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

  - Extremum location (offset)

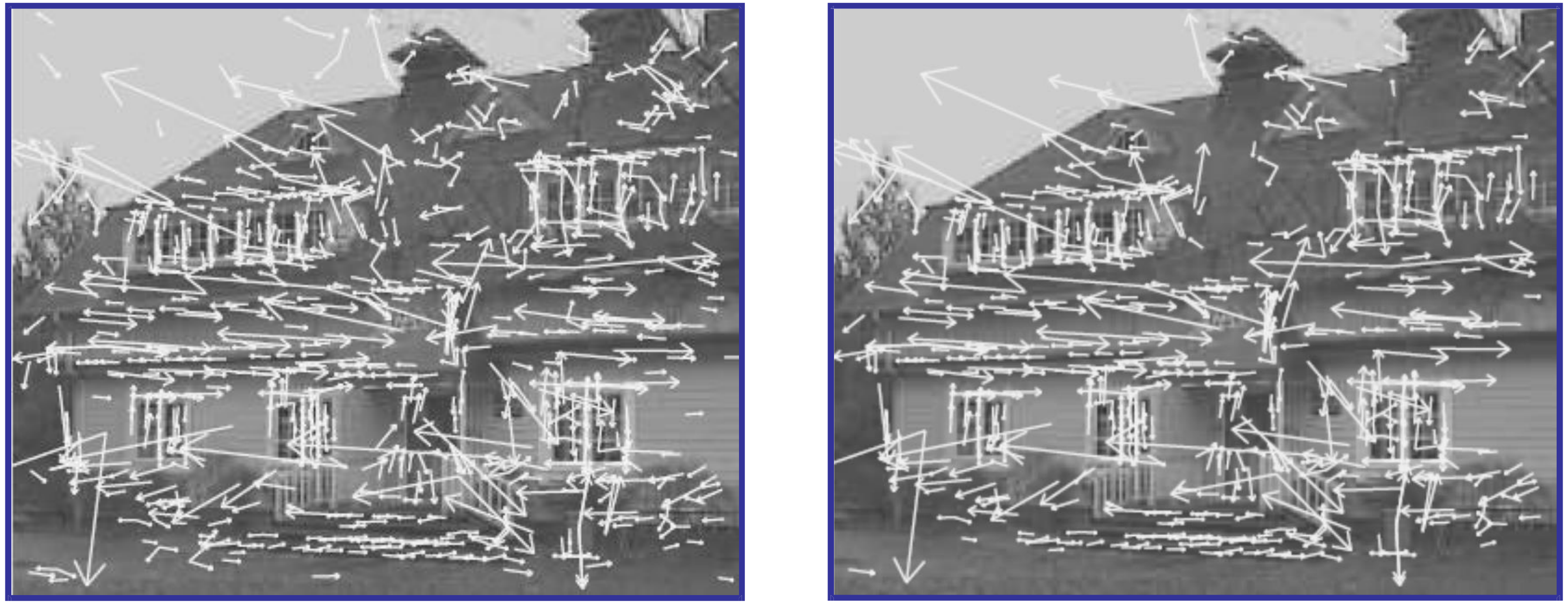
$$\hat{x} = -\frac{\partial^2 D^{-1}}{\partial x^2}\frac{\partial D}{x}$$

  - IF offset($\hat{x}$) is larger than 0.5
    → move the position of x



SIFT – Feature extraction

- Outlier removal #2 – Low contrast removal
  - $D(\hat{x}) > 0.03$



from 832 key points to 729 key points, th=0.03.

SIFT – Feature extraction

- Outlier removal #3 – Low curvature removal
  - Remeber, it's analogous to Harris corner detection

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

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

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

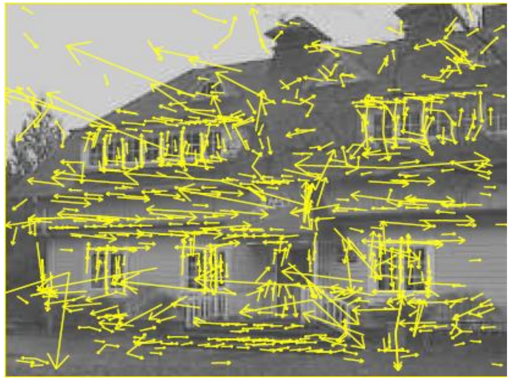
$$\frac{\mathrm{Tr}(\mathbf{H})^2}{\mathrm{Det}(\mathbf{H})} = \frac{(\alpha+\beta)^2}{\alpha\beta} = \frac{(r\beta+\beta)^2}{r\beta^2} = \frac{(r+1)^2}{r},$$

SIFT – Feature extraction

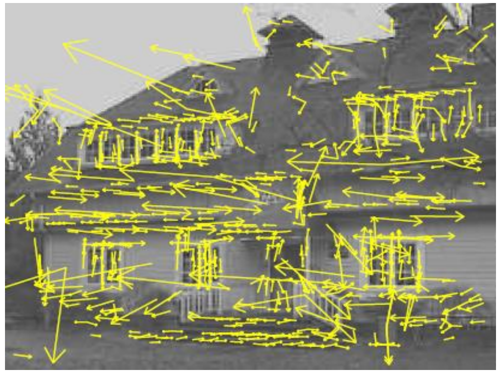
- Key point detection example

- Original image

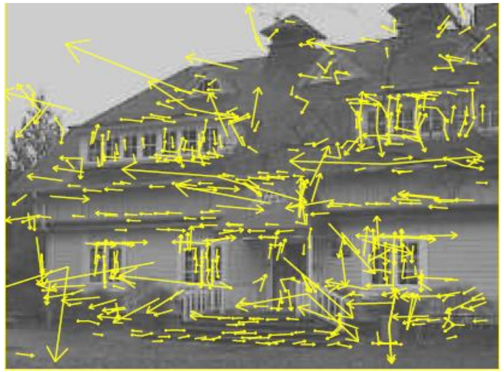
2. Initial features (832)



- Low contrast removed (729)



4. Low curvature removed (536)



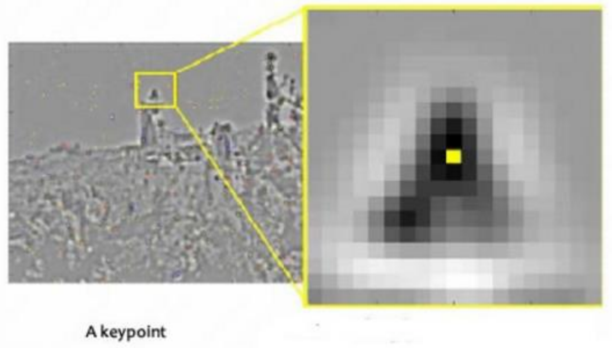
SIFT – Key point descriptor

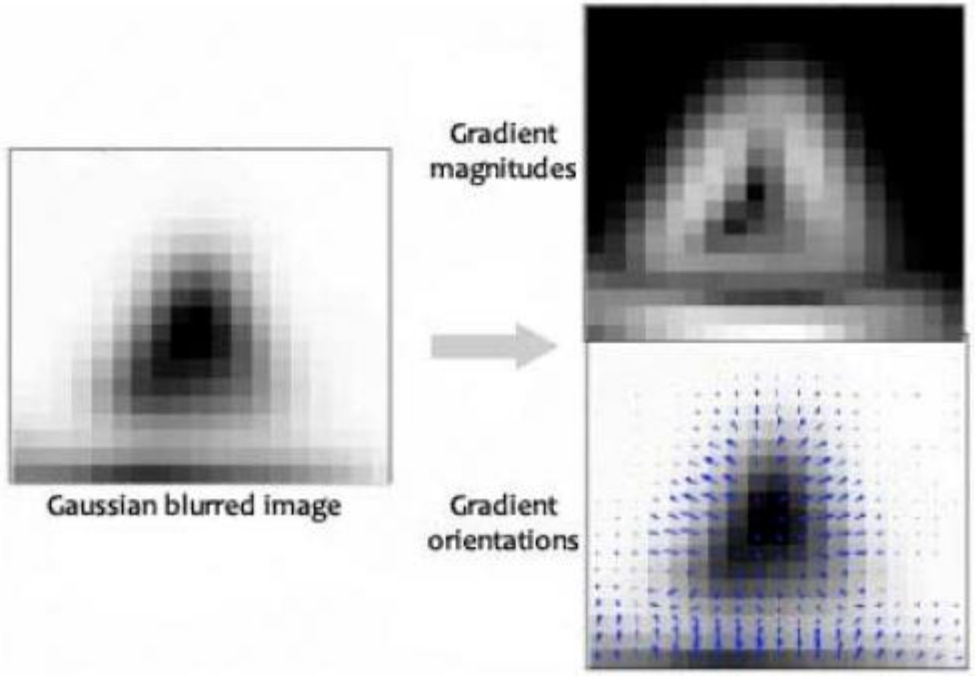
- Orientation assignment
  - Compute gradient magnitude and orientation for each SIFT point (x,y,σ) :

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

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

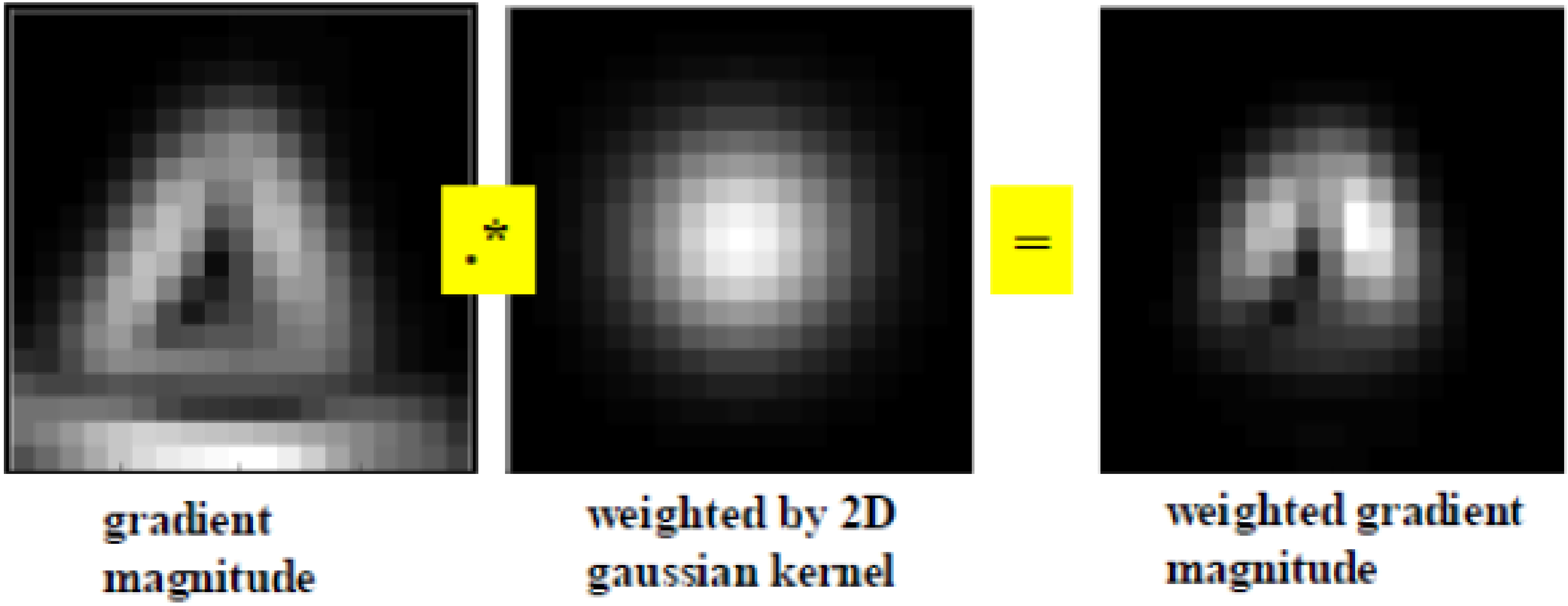
  - Compute gradient histogram





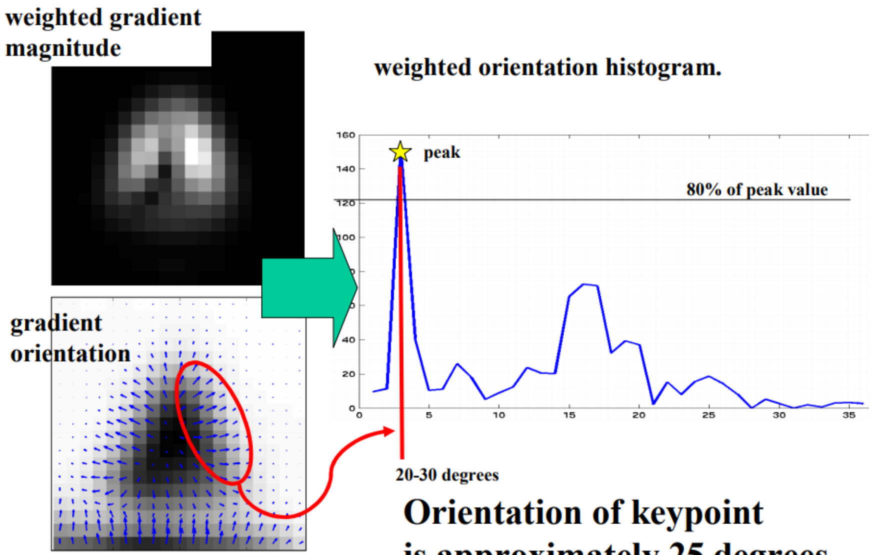
SIFT – Key point descriptor

- Weight with Gaussian function
  - To consider only pixels within the same distance.

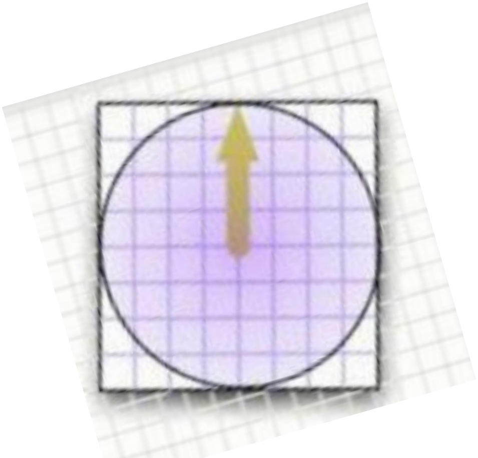


SIFT – Key point descriptor

- **Orientation assignment**
  - Create **histogram** of local gradient directions computed at selected scale
  - Assign **dominant orientation** at peak of smoothed histogram
  - Rotate image according to the direction → rotation invariant



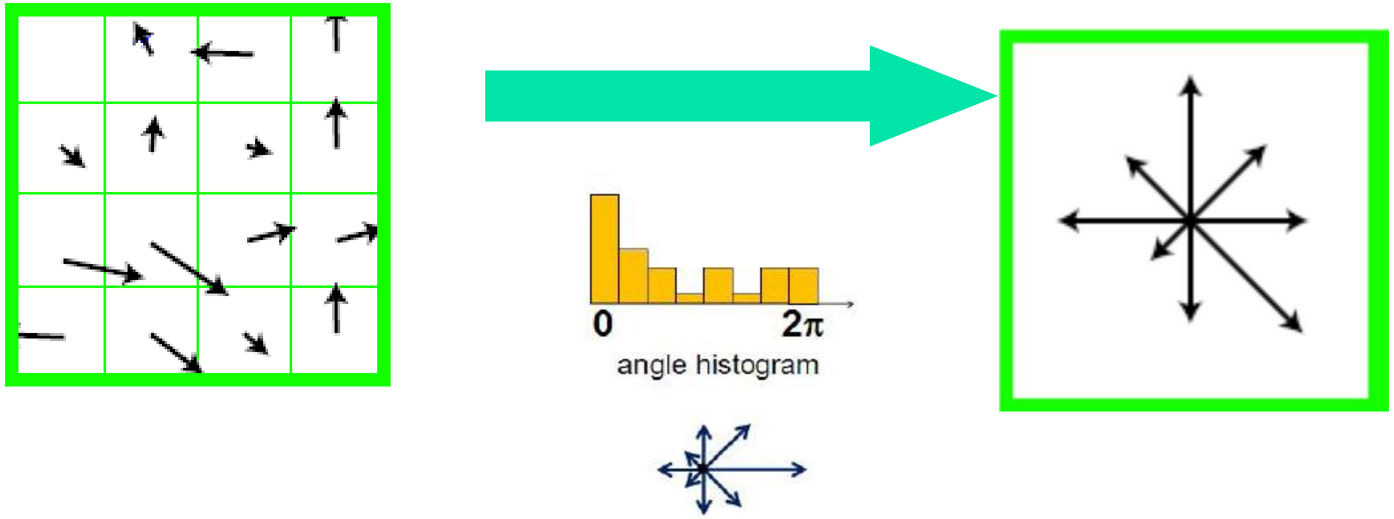
Orientation of keypoint is approximately 25 degrees



Rotate image

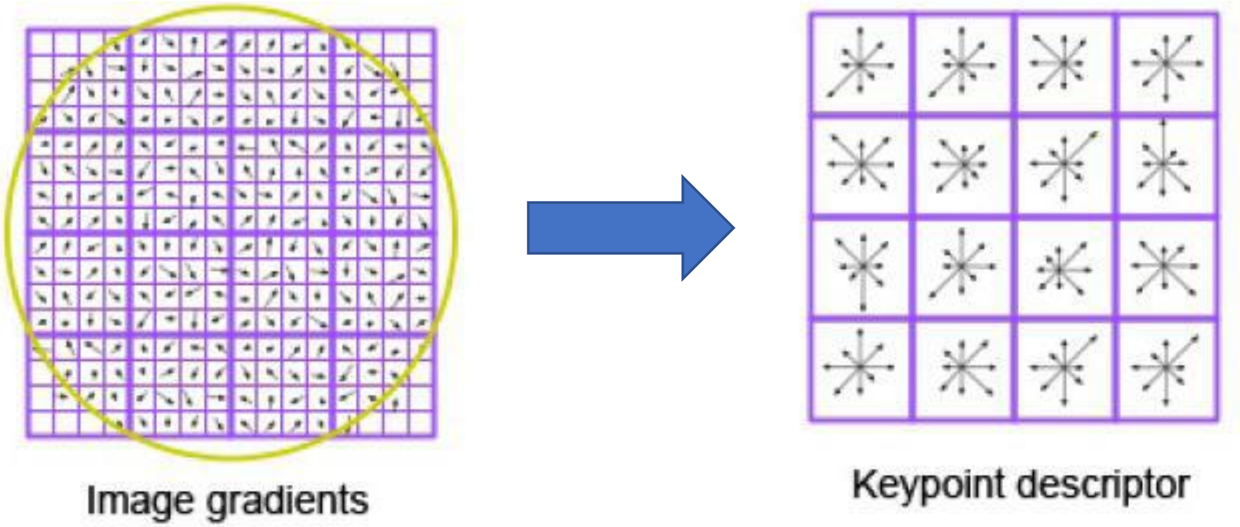
SIFT – Key point descriptor

- Orientation assignment like HOG method
  - 4x4 Gradient windows relative to key point orientation
  - **Histogram of 4x4 samples per window in 8 directions**



SIFT – Key point descriptor

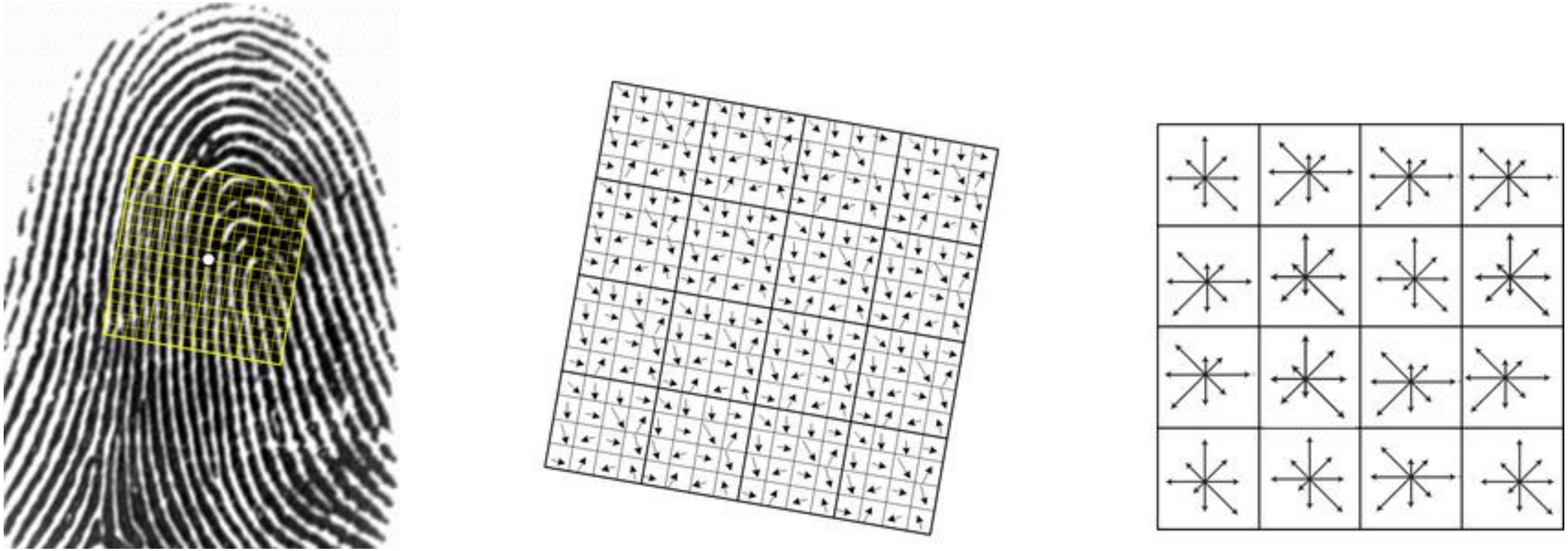
- Orientation assignment
  - Compute relative orientation and magnitude in a 16x16 neighborhood at key point
  - Form weighted histogram (8 bin) for 4x4 regions
    - Weight by magnitude and spatial Gaussian
    - Concatenate 16 histograms in one long vector of 128 dimensions



16 histograms x 8 orientations = 128 features

SIFT – Key point descriptor

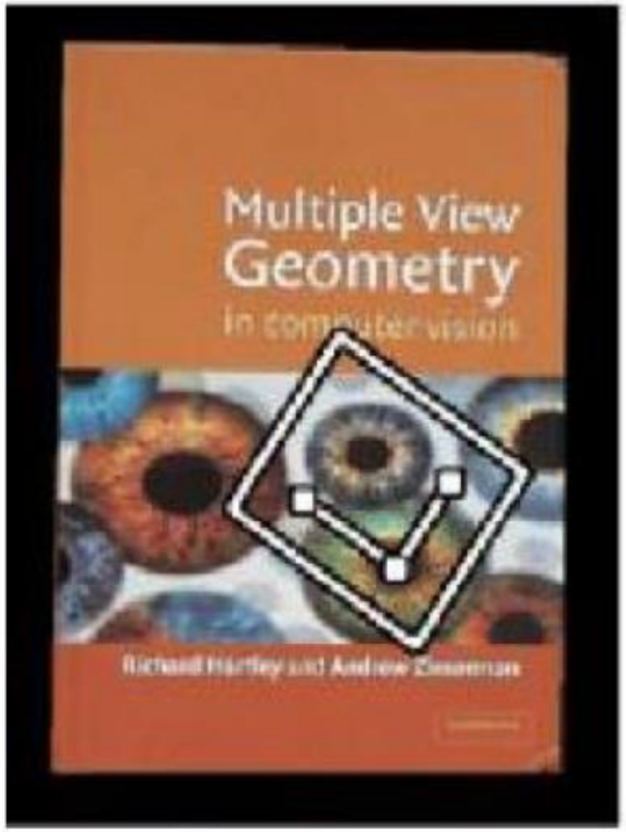
- Orientation assignment



https://www.researchgate.net/figure/The-process-of-building-a-single-SIFT-keypoint-descriptor-a-A-Single-SIFT-keypoint_fig6_237090165

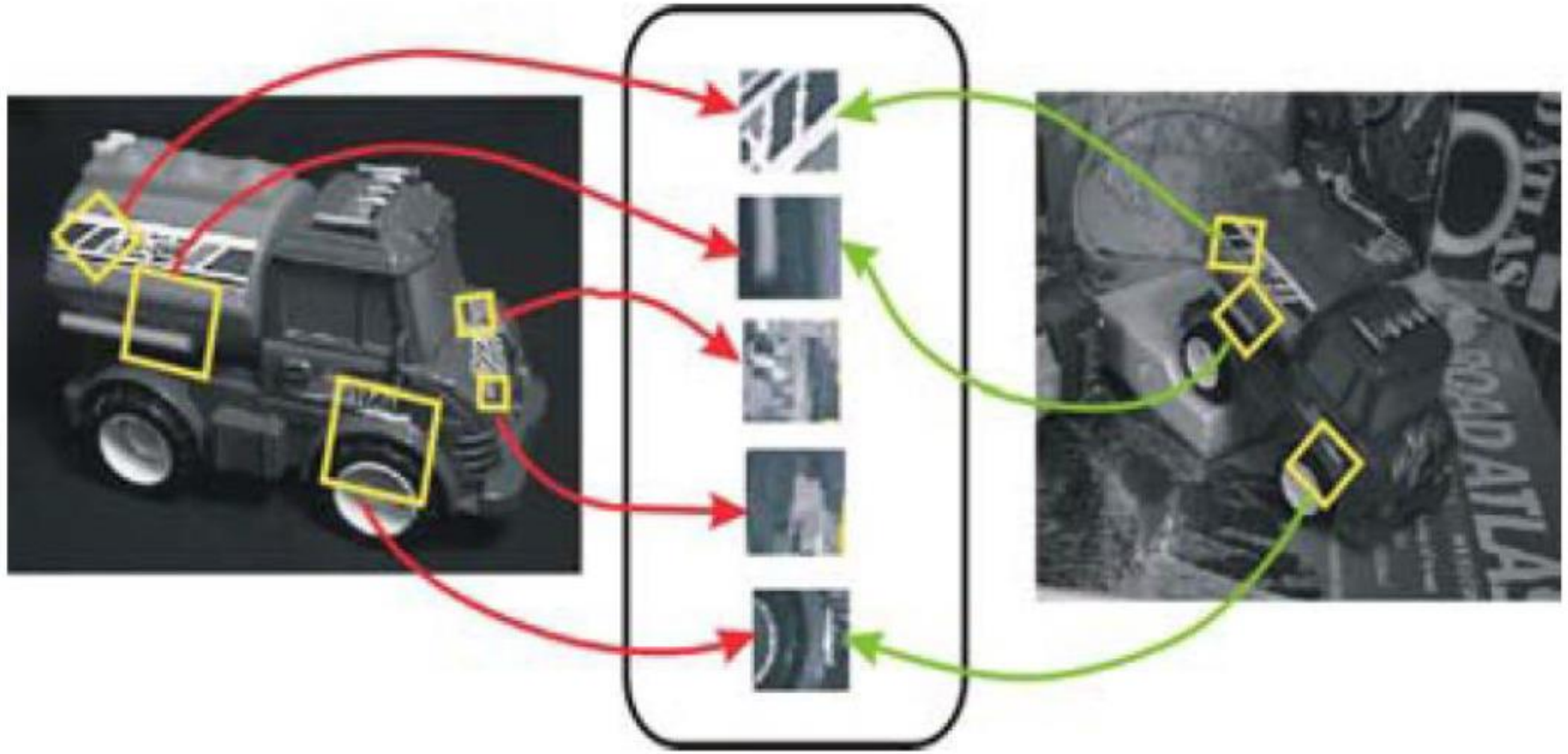
SIFT – Key point descriptor

- Scale and Rotation invariance



SIFT – Key point descriptor

- Scale and Rotation invariance
- But not invariant to affine transform



From https://courses.cs.washington.edu/courses/cse455/08wi/lectures/features.pdf

Thank you