



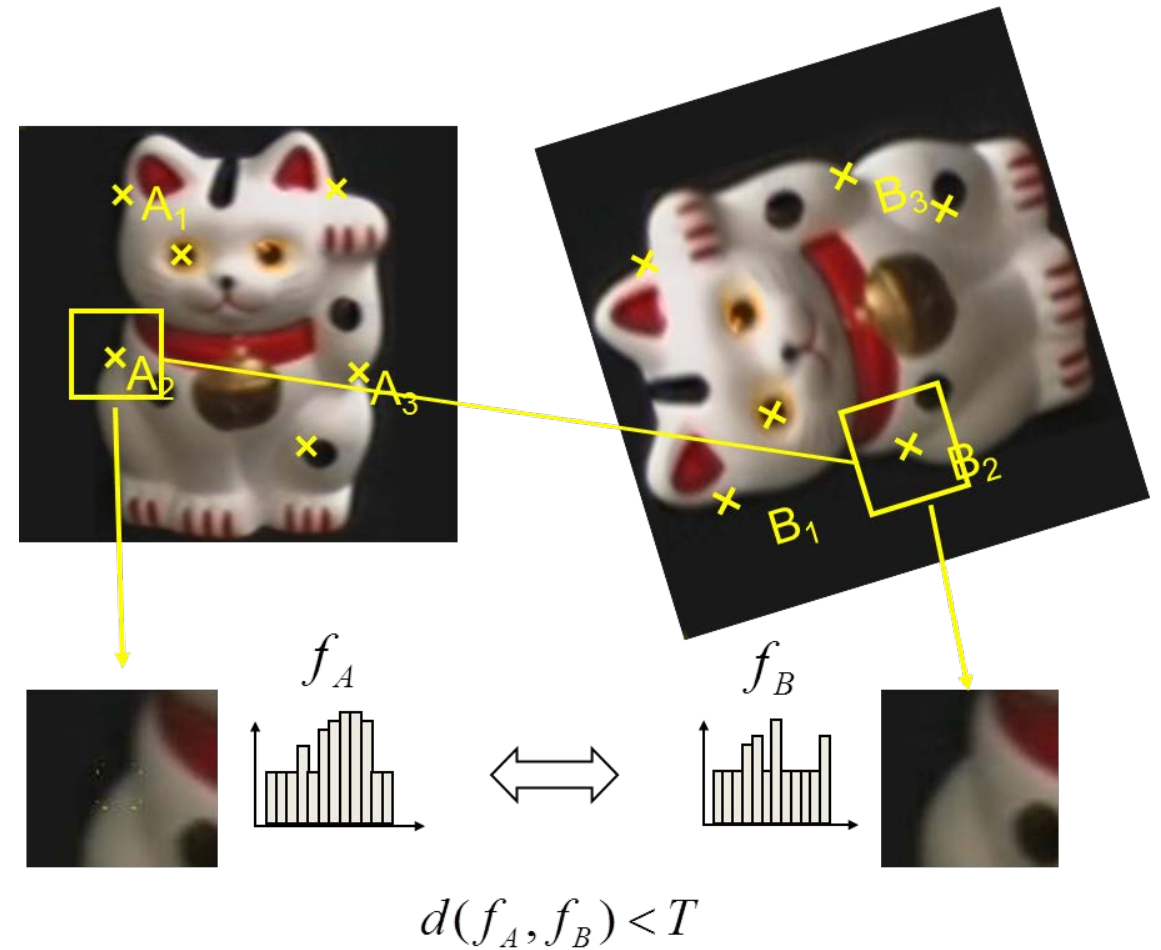
# Lecture 4.1

## Feature descriptors

Trym Vegard Haavardsholm

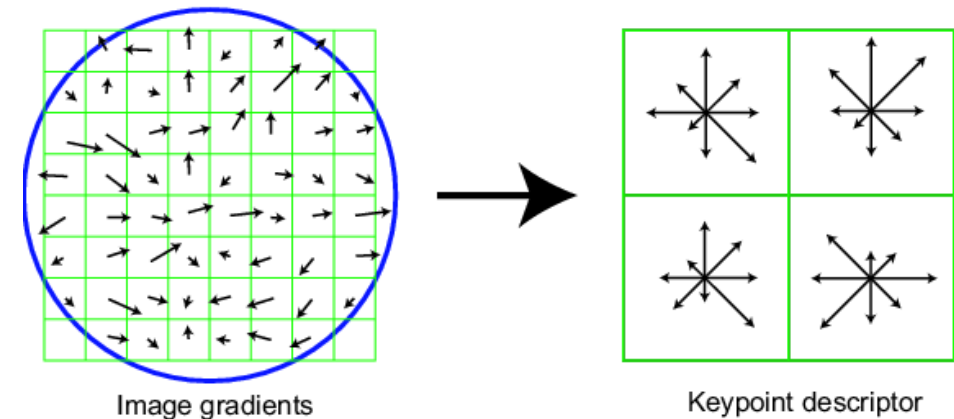
# Feature descriptors

- Histogram of Gradients (HoG) descriptors
- Binary descriptors

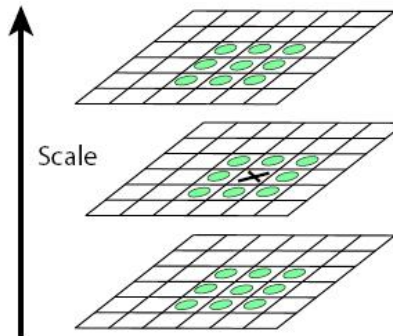
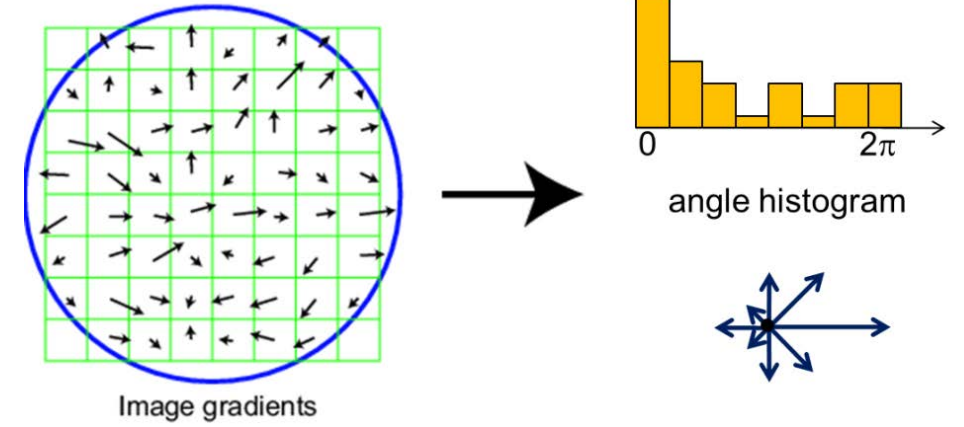
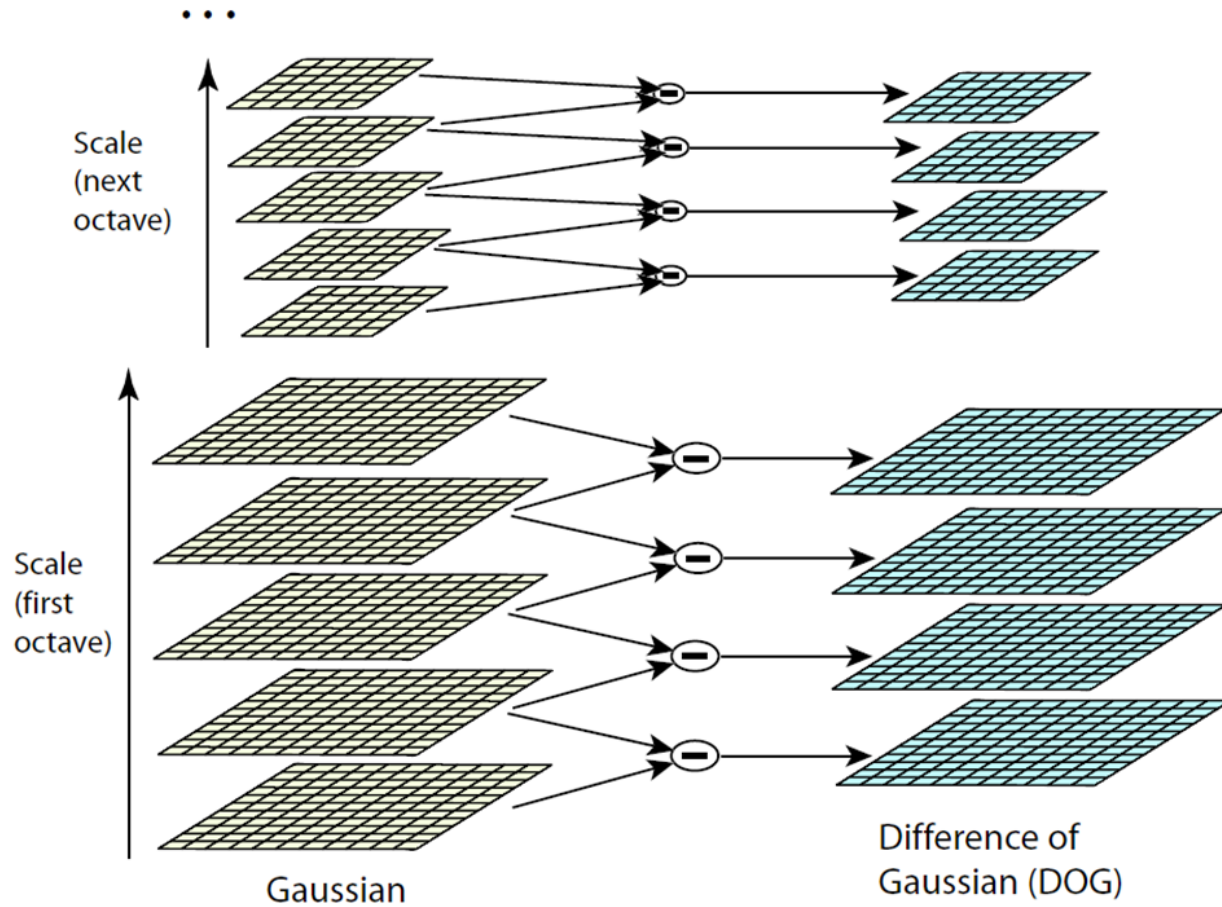


# Histogram of Gradients (HOG) descriptors

- Scale Invariant Feature Transform (SIFT)
  - David G. Lowe.  
["Distinctive image features from scale-invariant keypoints."](#)  
*IJCV* 60 (2), pp. 91-110, 2004
- Speeded Up Robust Features (SURF)
  - Bay, Herbert, Tinne Tuytelaars, and Luc Van Gool.  
["Surf: Speeded up robust features."](#)  
*Computer Vision–ECCV 2006*.  
Springer Berlin Heidelberg, 2006. 404-417



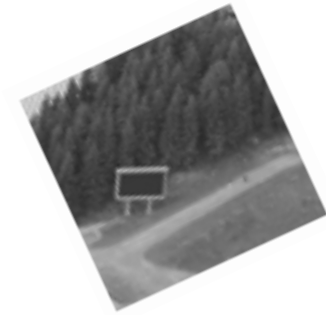
# SIFT detector



# SIFT detections

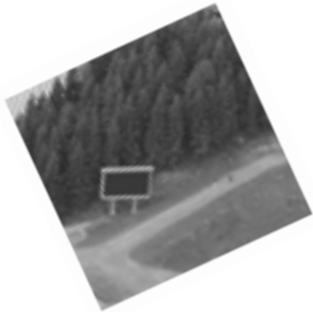


# Patch at detected position, scale, orientation



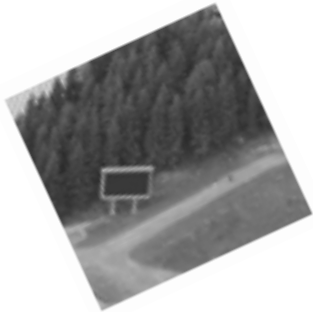
# SIFT descriptor

- Extract patch around detected keypoint
- Normalize the patch to canonical scale and orientation



# SIFT descriptor

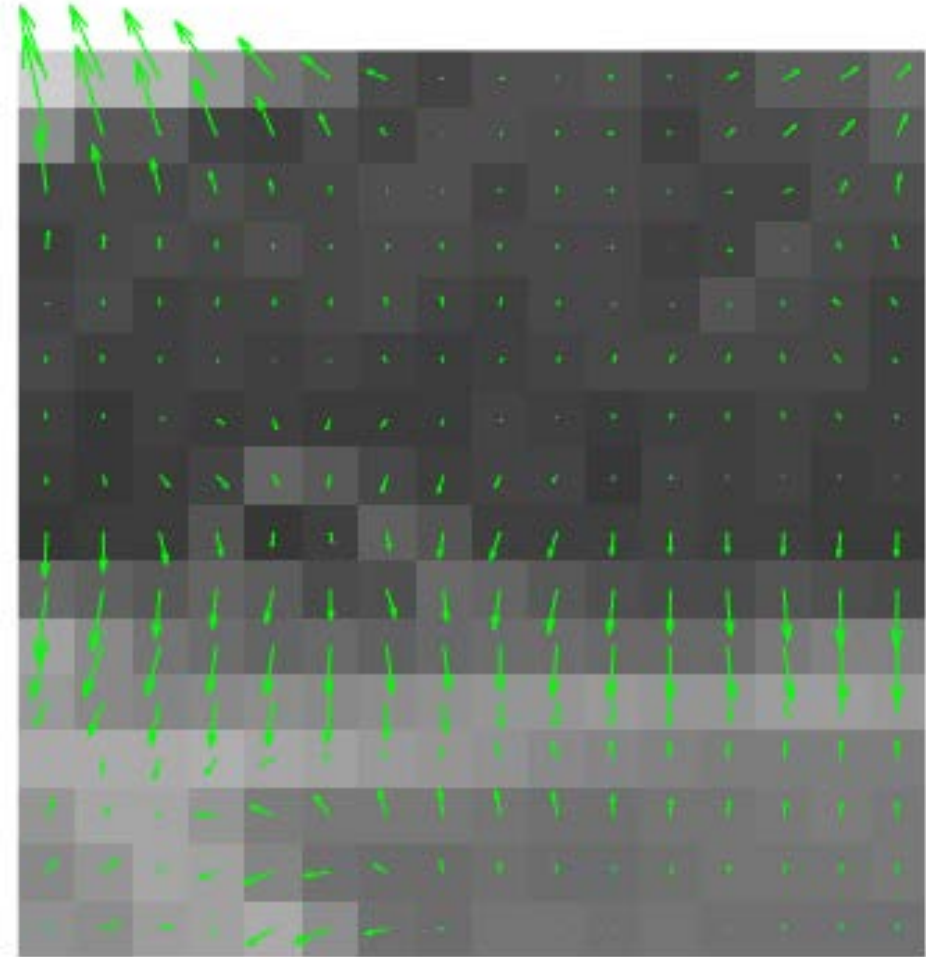
- Extract patch around detected keypoint
- Normalize the patch to canonical scale and orientation
- Resize patch to 16x16 pixels





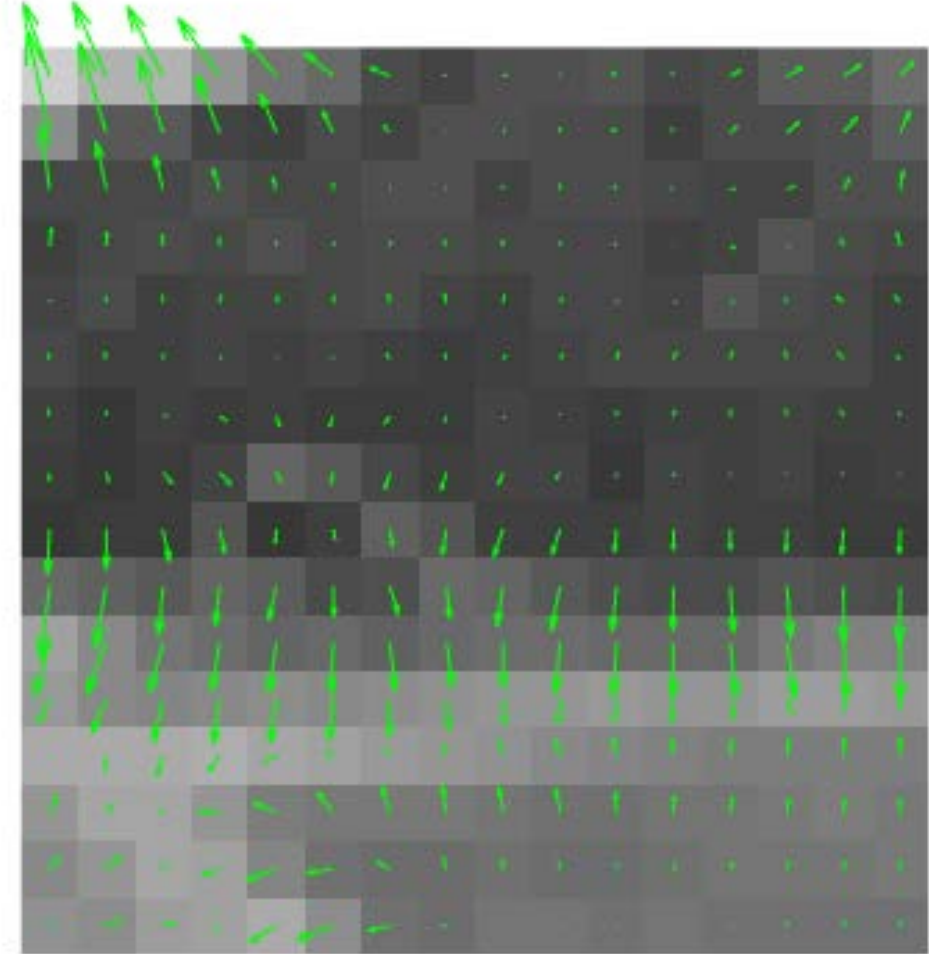
# SIFT descriptor

- Compute the gradients



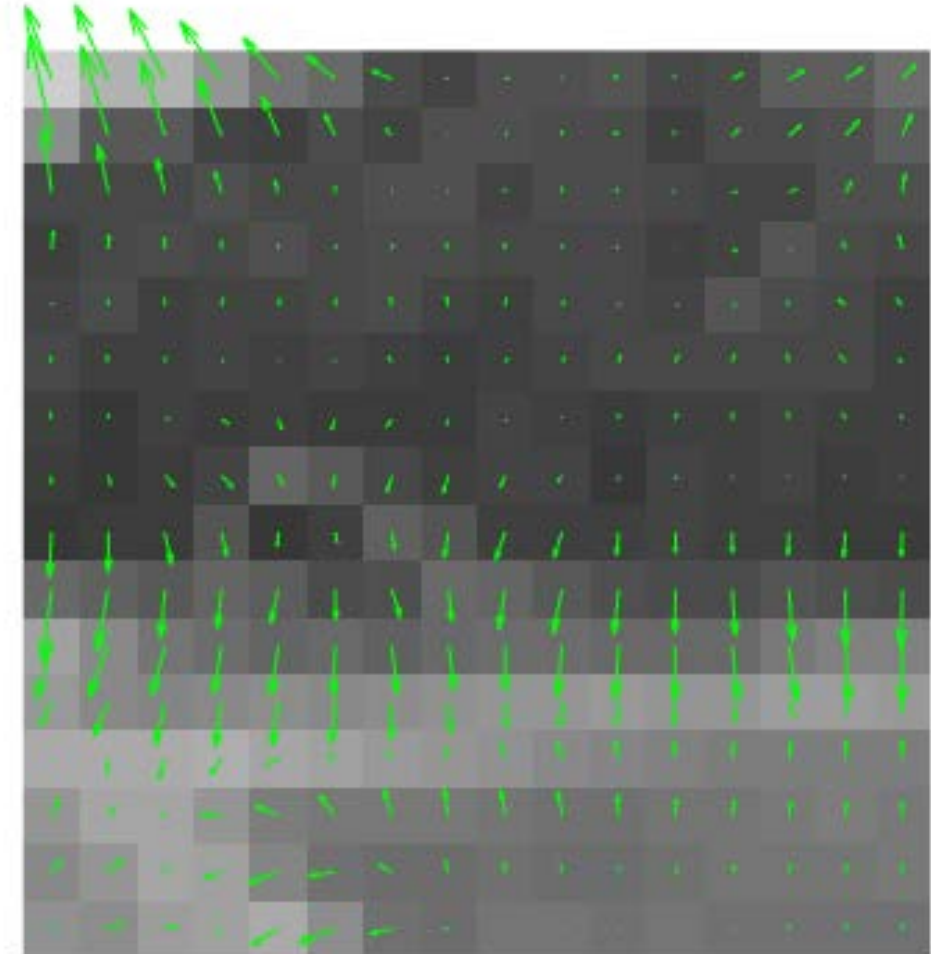
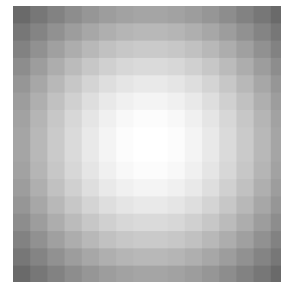
# SIFT descriptor

- Compute the gradients
  - Unaffected by additive intensity change



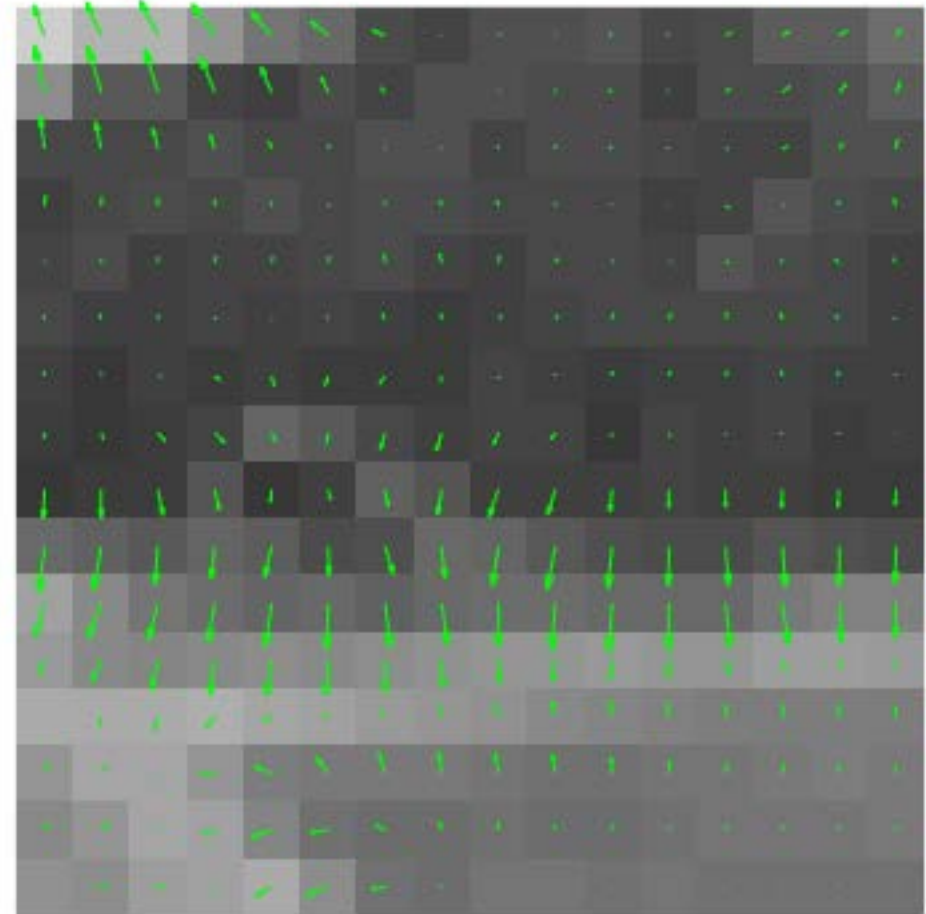
# SIFT descriptor

- Compute the gradients
  - Unaffected by additive intensity change
- Apply a Gaussian weighting function



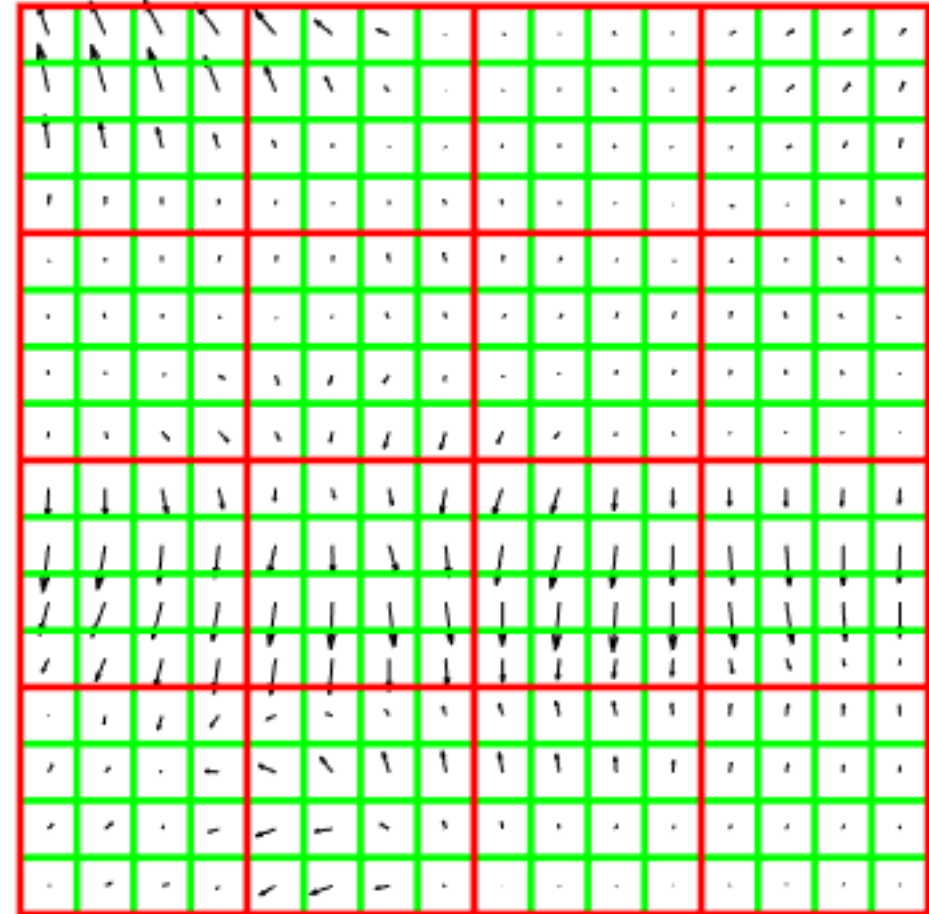
# SIFT descriptor

- Compute the gradients
  - Unaffected by additive intensity change
- Apply a Gaussian weighting function
  - Weighs down gradients far from the centre
  - Avoids sudden changes in the descriptor with small changes in the window position



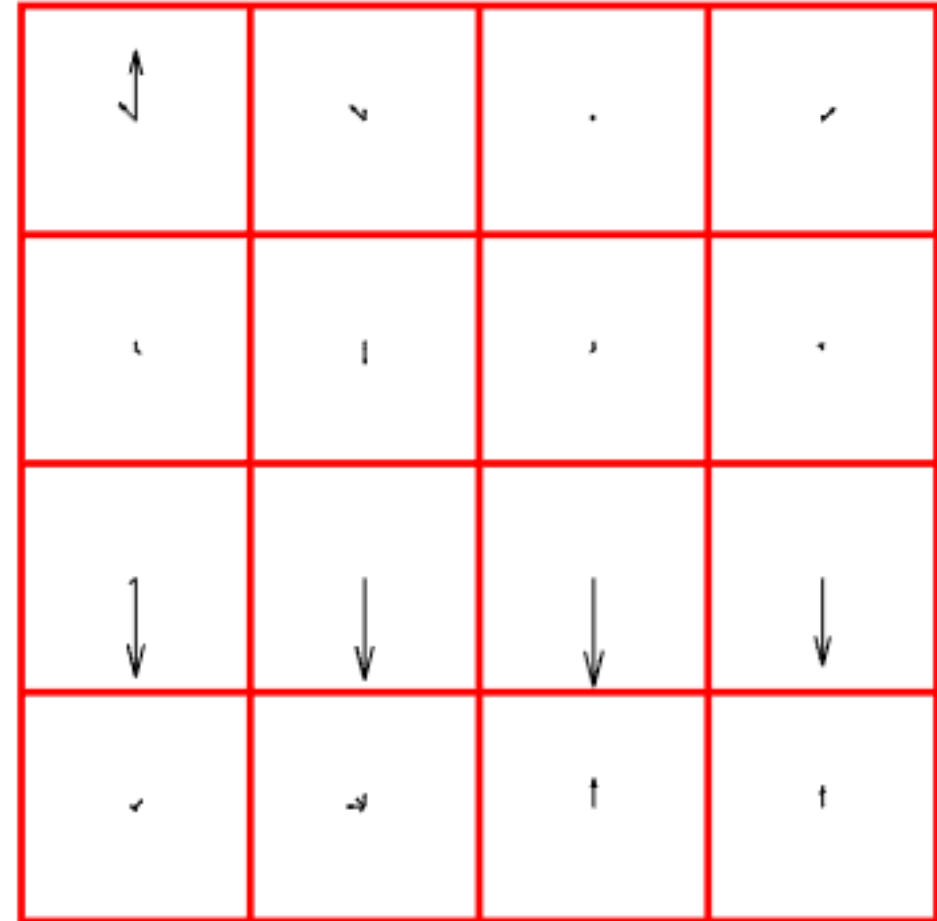
# SIFT descriptor

- Compute the gradients
  - Unaffected by additive intensity change
- Apply a Gaussian weighting function
  - Weighs down gradients far from the centre
  - Avoids sudden changes in the descriptor with small changes in the window position
- Divide the patch into 16 4x4 pixels squares



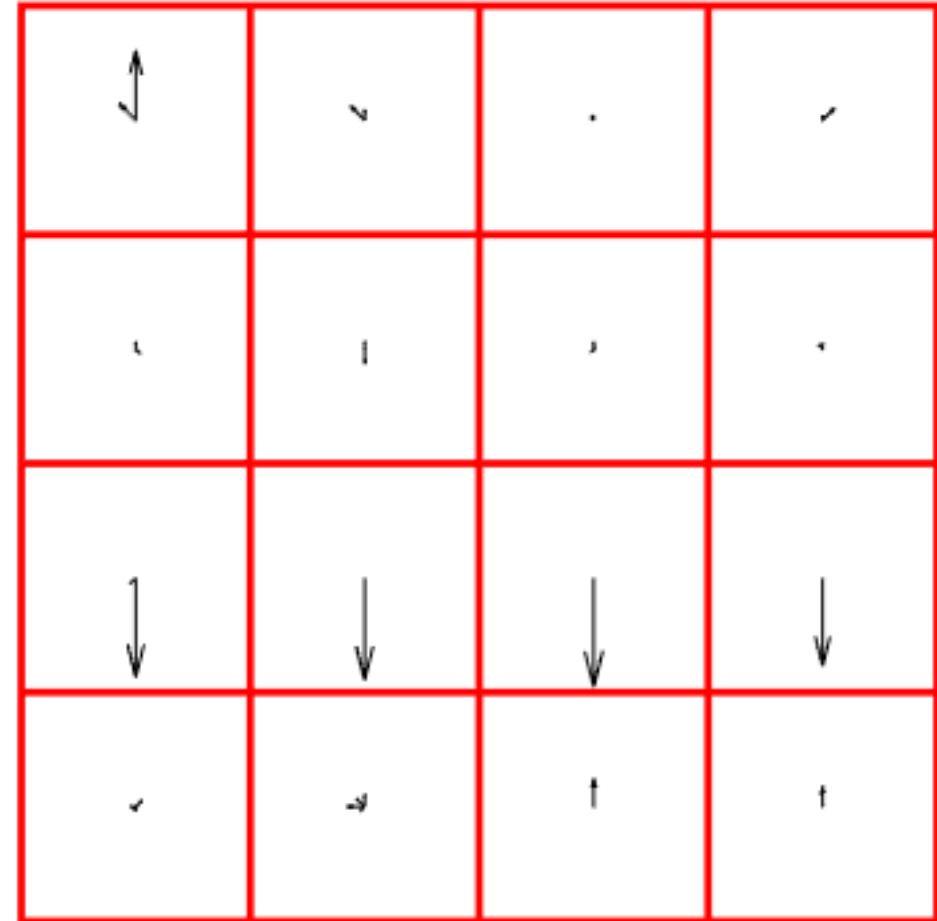
# SIFT descriptor

- Compute gradient direction histograms over 8 directions in each square



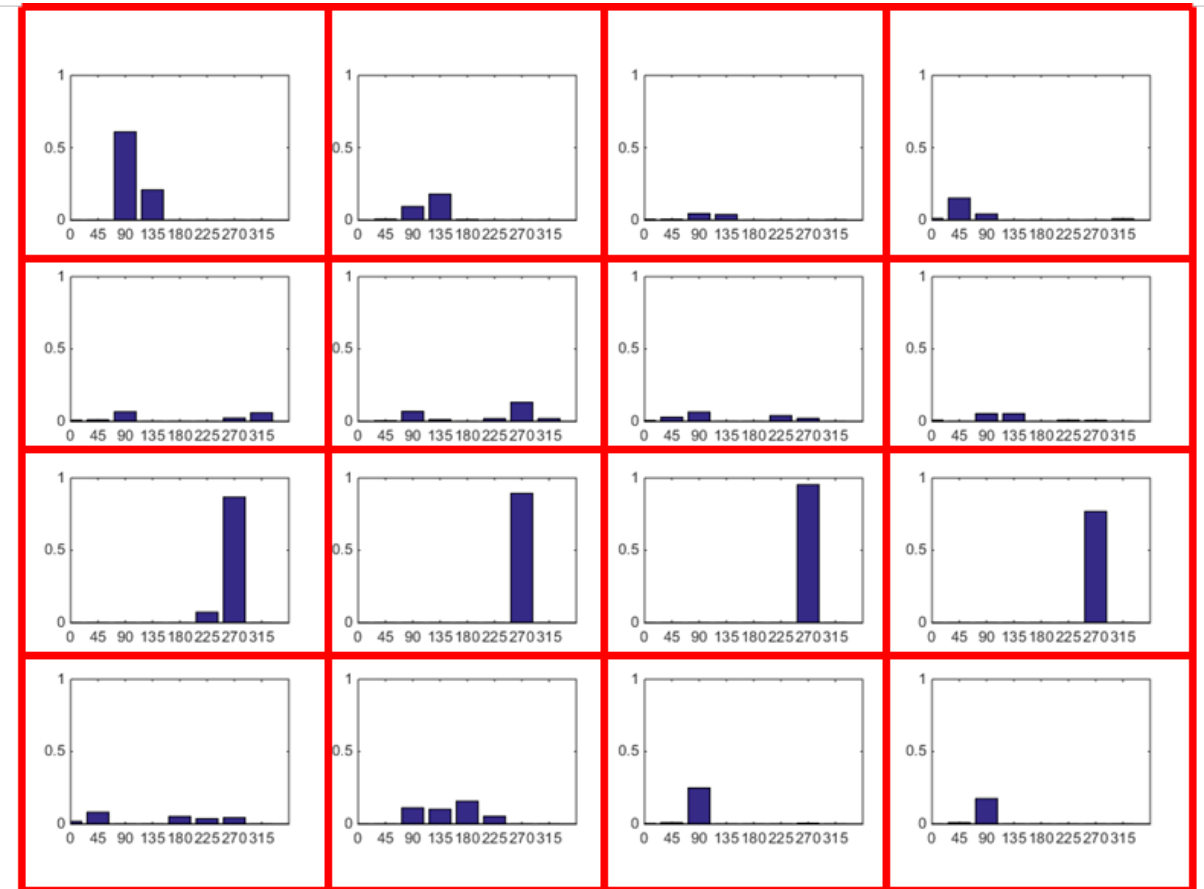
# SIFT descriptor

- Compute gradient direction histograms over 8 directions in each square
  - Trilinear interpolation
  - Robust to small shifts, while preserving some spatial information



# SIFT descriptor

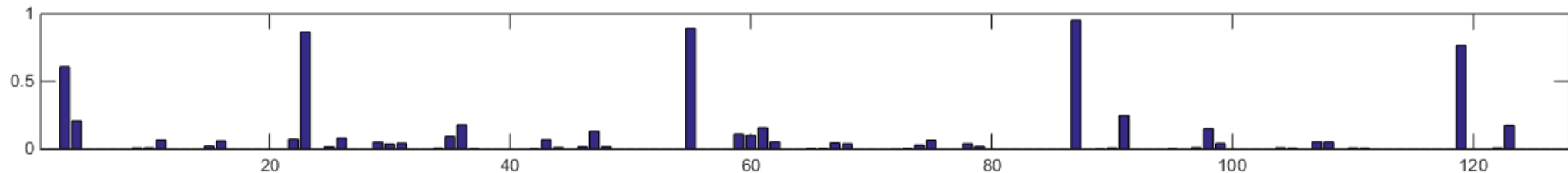
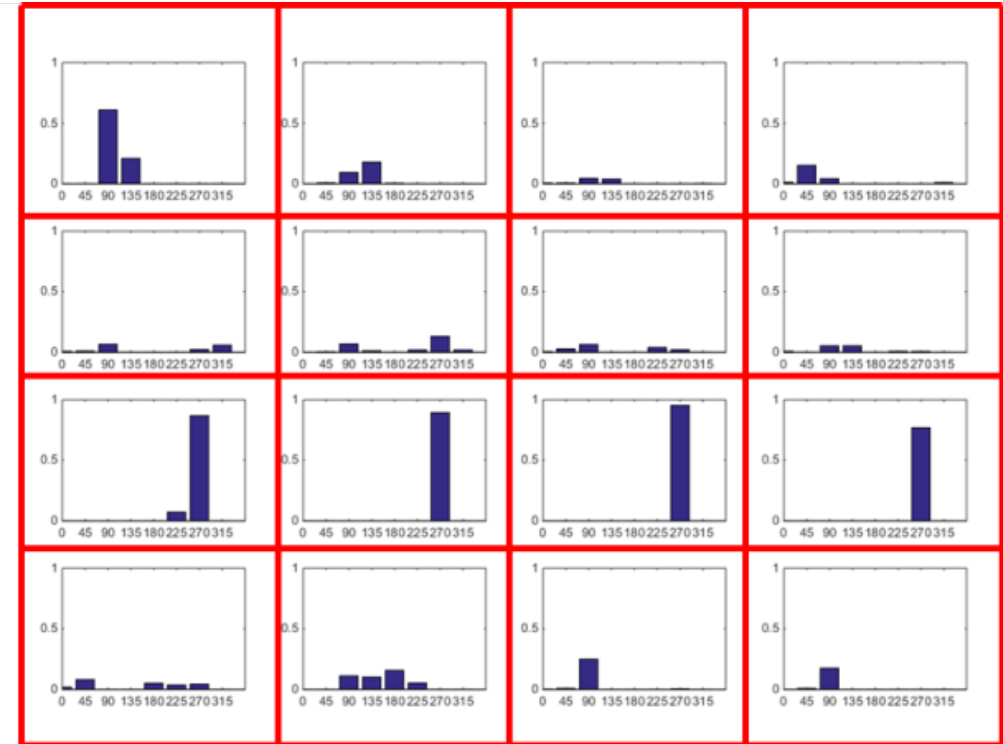
- Compute gradient direction histograms over 8 directions in each square
  - Trilinear interpolation
  - Robust to small shifts, while preserving some spatial information





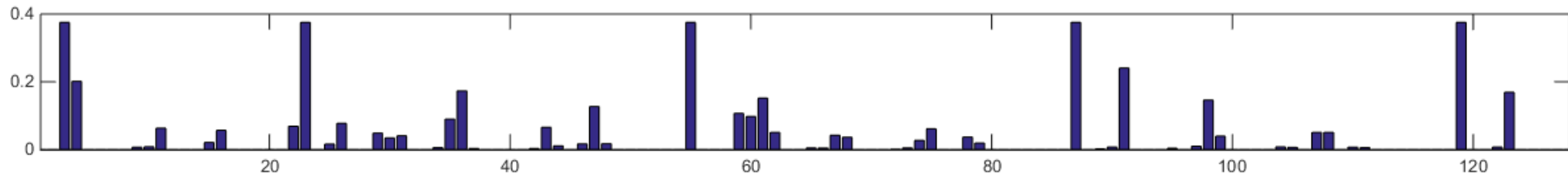
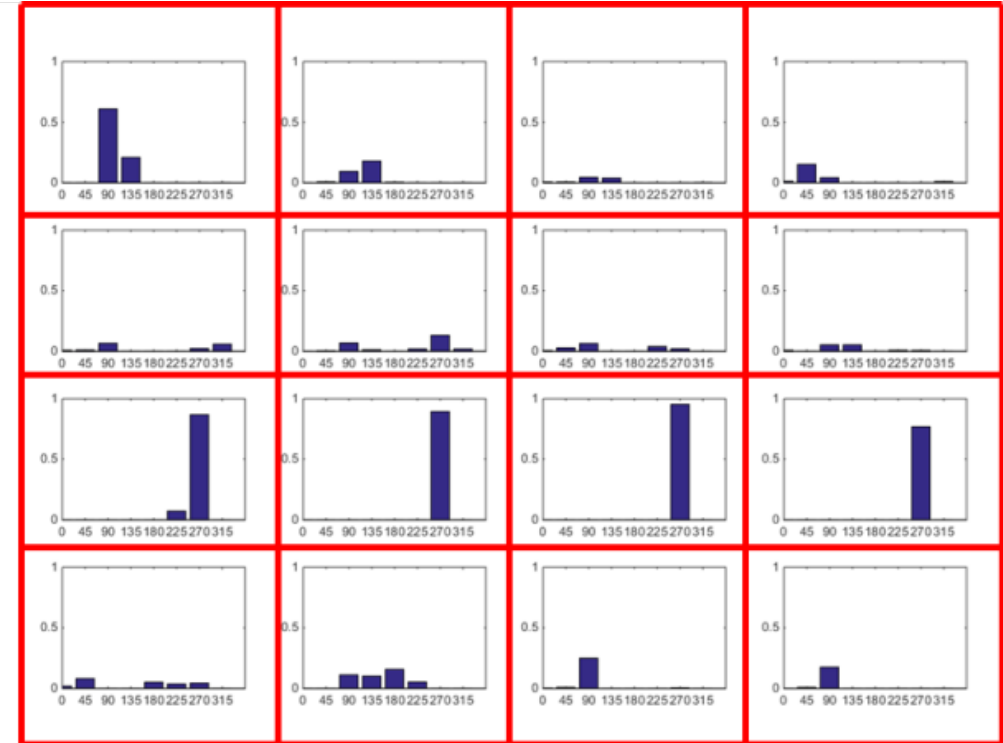
# SIFT descriptor

- Concatenate the histograms to obtain a 128 dimensional feature vector



# SIFT descriptor

- Concatenate the histograms to obtain a 128 dimensional feature vector
- Normalize to unit length
  - Invariant to multiplicative contrast change
- Threshold gradient magnitudes to avoid excessive influence of high gradients
  - Clamp gradients  $> 0.2$
  - Renormalize



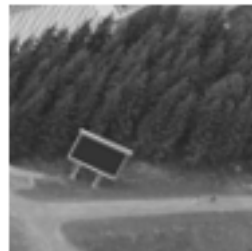
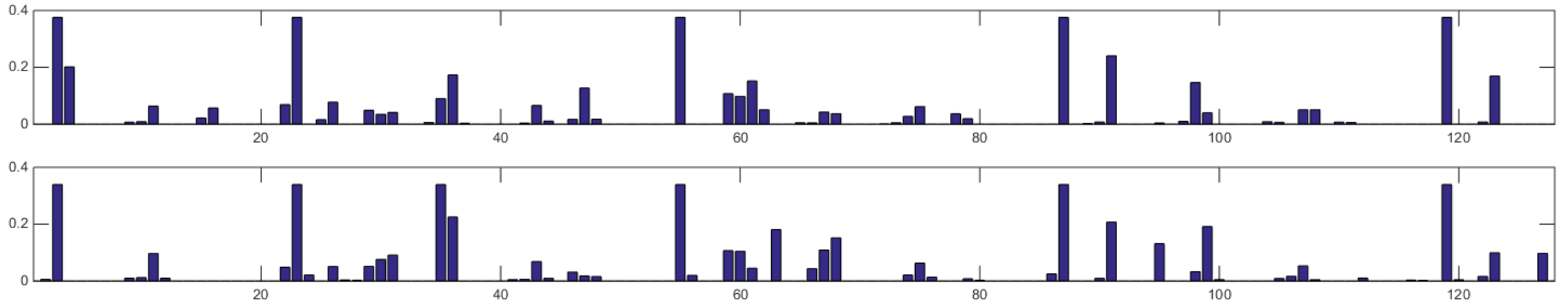
# Example: Feature comparison



# Example: Feature comparison

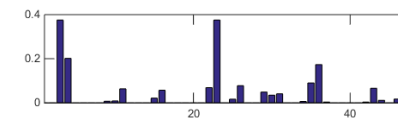
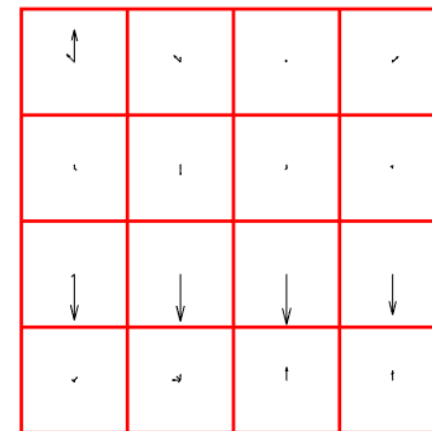
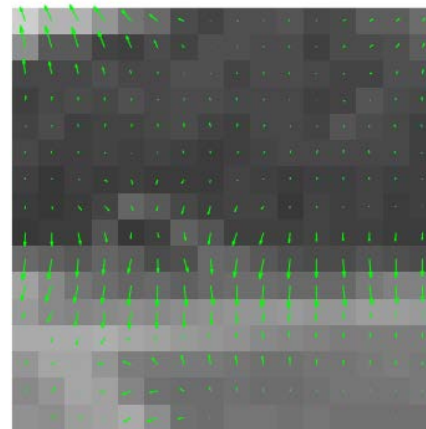
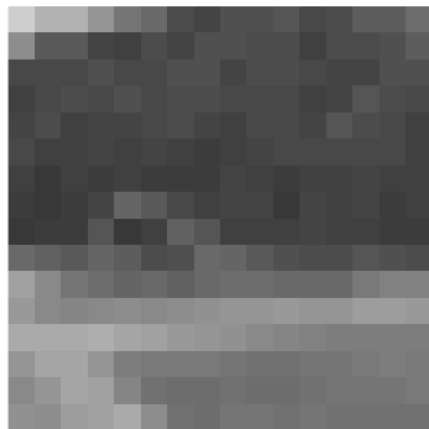
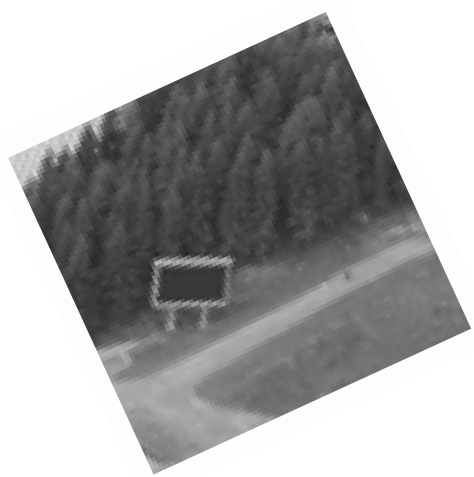


# Example: Feature comparison

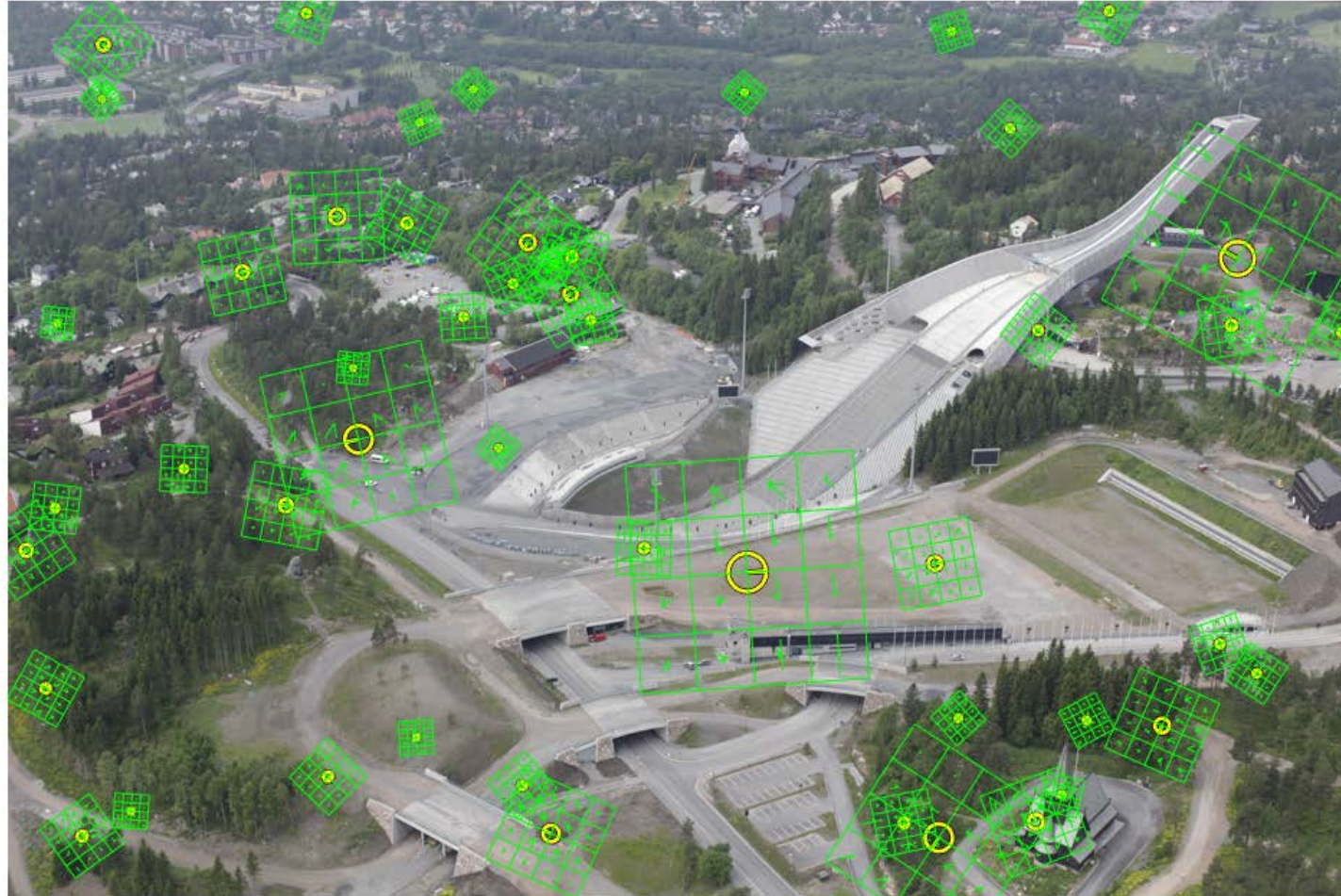


# SIFT summary

- Extract a 16x16 patch around detected keypoint
- Compute the gradients and apply a Gaussian weighting function
- Divide the window into a 4x4 grid of cells
- Compute gradient direction histograms over 8 directions in each cell
- Concatenate the histograms to obtain a 128 dimensional feature vector
- Normalize to unit length



# SIFT summary



# Binary descriptors

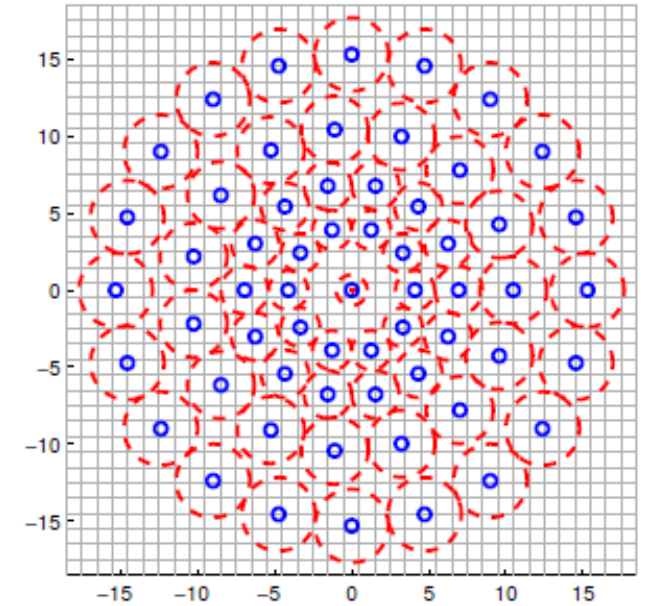
- Extremely efficient construction and comparison
- Based on pairwise intensity comparisons
  - Sampling pattern around keypoint
  - Set of sampling pairs
  - Feature descriptor vector is a binary string:

$$F = \sum_{0 \leq a \leq N} 2^a T(P_a)$$

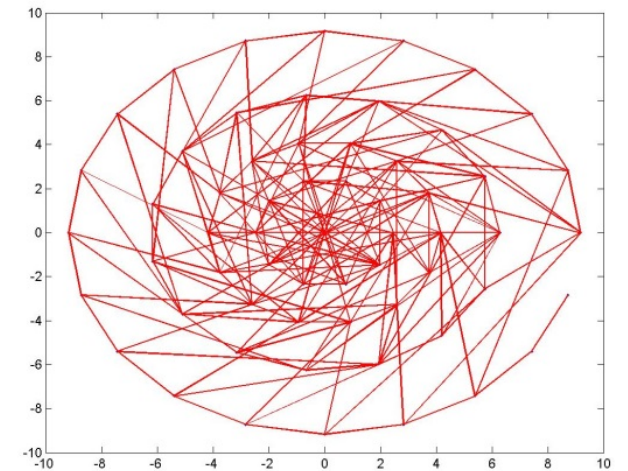
$$T(P_a) = \begin{cases} 1 & \text{if } I(P_a^{r1}) > I(P_a^{r2}) \\ 0 & \text{otherwise} \end{cases}$$

- Matching using Hamming distance:

$$L = \sum_{0 \leq a \leq N} XOR(F_a^1, F_a^2)$$



BRISK sampling pattern

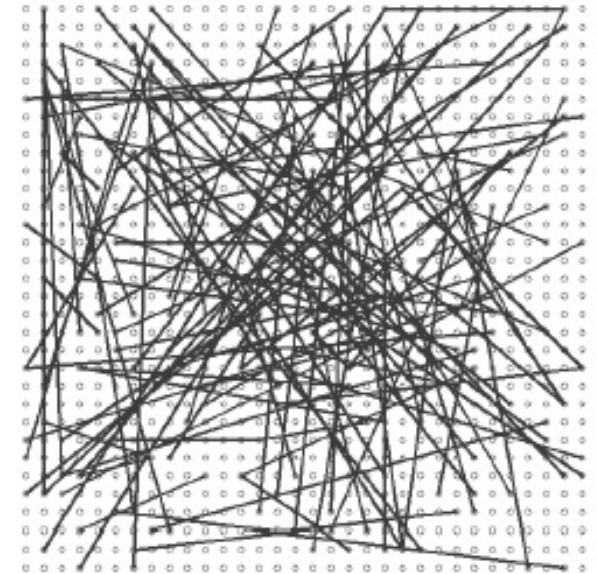


BRISK sampling pairs



# Binary descriptors

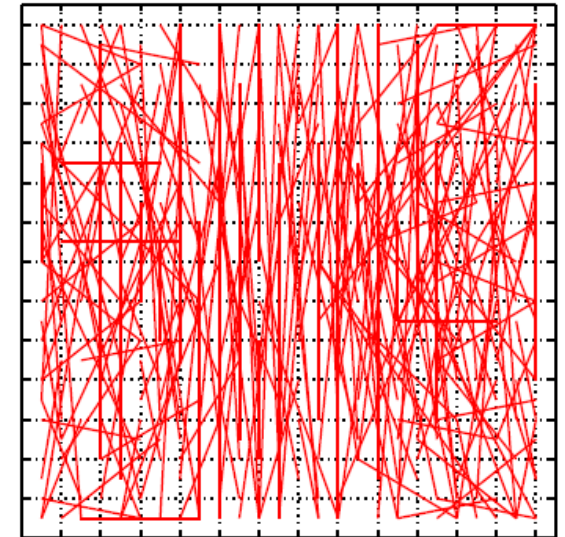
Method	Sampling pattern	Orientation calculation	Sampling pairs
BRIEF	None	None	Random
ORB	None	Moments	Learned pairs
BRISK	Concentric circles, More points on outer rings	Comparing gradients of long pairs	Short pairs
FREAK	Overlapping concentric circles, more points on inner rings	Comparing gradients of preselected 45 pairs	Learned pairs



BRIEF sampling pairs

# Binary descriptors

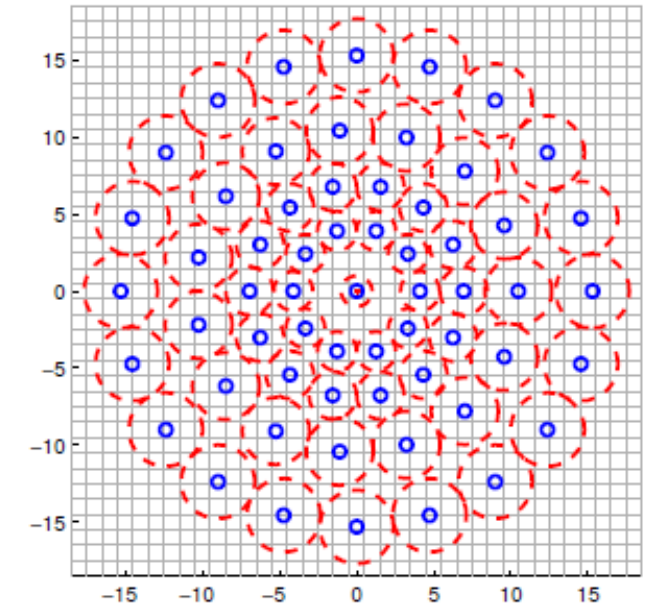
Method	Sampling pattern	Orientation calculation	Sampling pairs
BRIEF	None	None	Random
<b>ORB</b>	None	Moments	Learned pairs
BRISK	Concentric circles, More points on outer rings	Comparing gradients of long pairs	Short pairs
FREAK	Overlapping concentric circles, more points on inner rings	Comparing gradients of preselected 45 pairs	Learned pairs



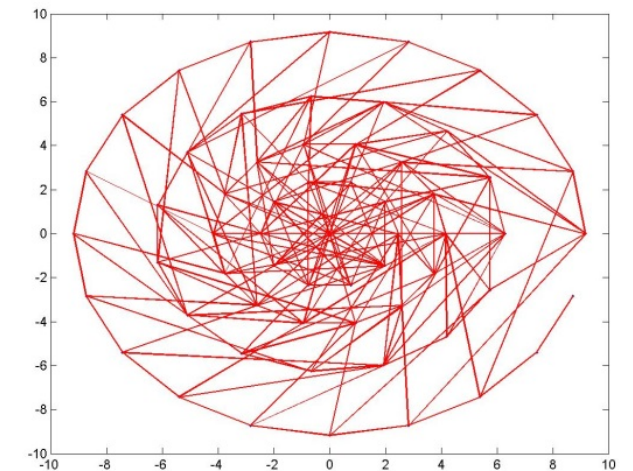
ORB sampling pairs

# Binary descriptors

Method	Sampling pattern	Orientation calculation	Sampling pairs
BRIEF	None	None	Random
ORB	None	Moments	Learned pairs
<b>BRISK</b>	Concentric circles, More points on outer rings	Comparing gradients of long pairs	Short pairs
FREAK	Overlapping concentric circles, more points on inner rings	Comparing gradients of preselected 45 pairs	Learned pairs



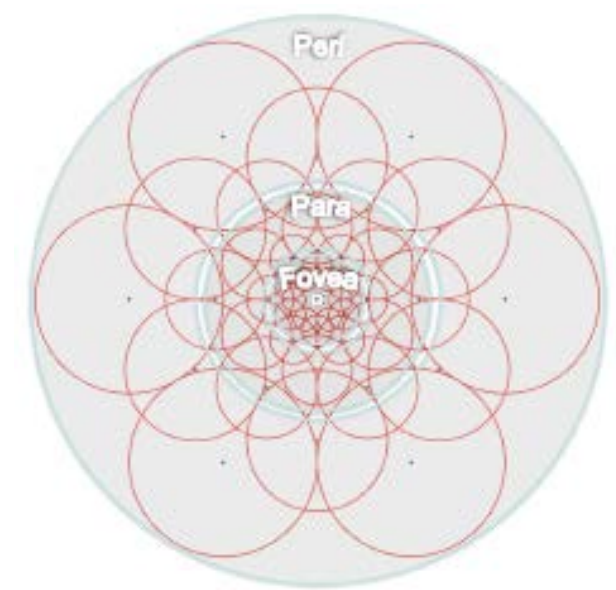
BRISK sampling pattern



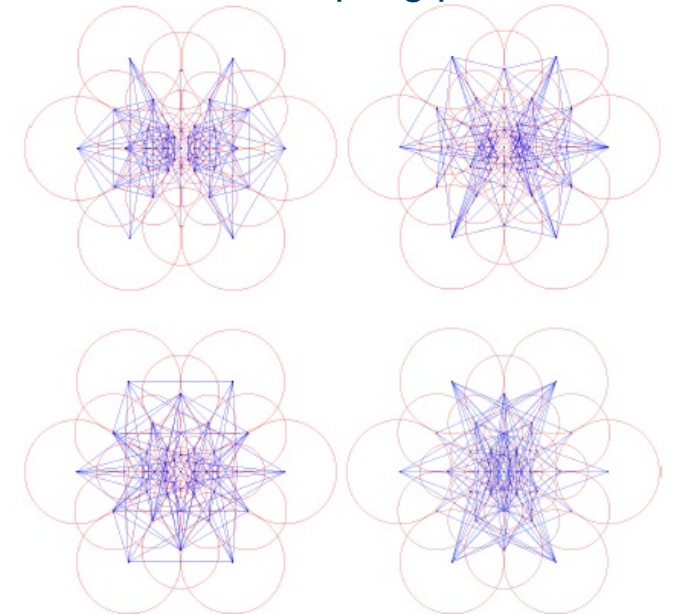
BRISK sampling pairs

# Binary descriptors

Method	Sampling pattern	Orientation calculation	Sampling pairs
BRIEF	None	None	Random
ORB	None	Moments	Learned pairs
BRISK	Concentric circles, More points on outer rings	Comparing gradients of long pairs	Short pairs
<b>FREAK</b>	Overlapping concentric circles, more points on inner rings	Comparing gradients of preselected 45 pairs	Learned pairs



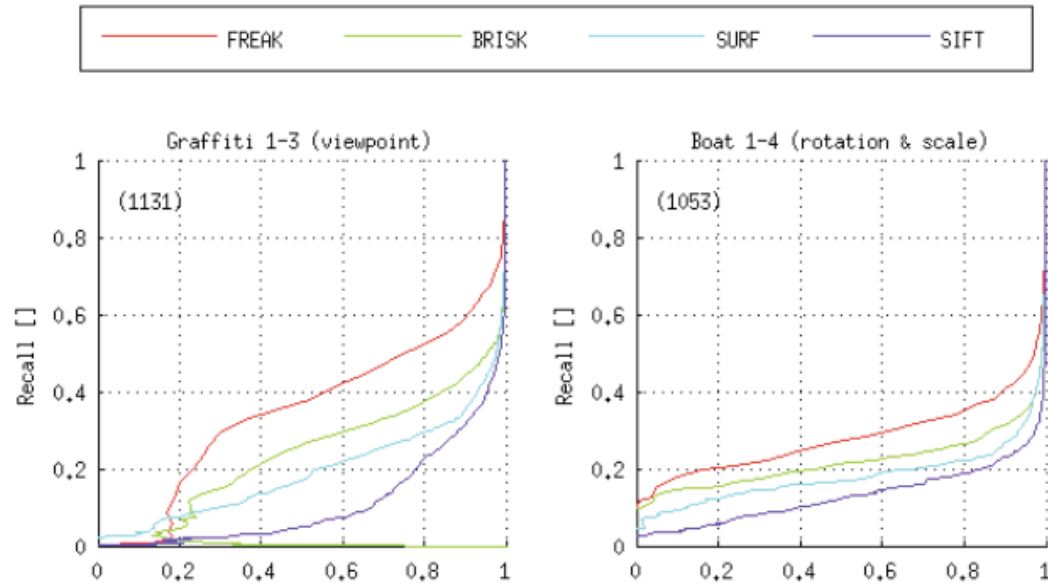
FREAK sampling pattern



FREAK sampling pairs

# Binary descriptors

- Often achieves very good performance compared to SIFT/SURF
- Much faster than SIFT/SURF



Time per keypoint	SIFT	SURF	BRISK	FREAK
Description in [ms]	2.5	1.4	0.031	0.018
Matching time in [ns]	1014	566	36	25

**Table 1:** Computation time on 800x600 images where approximately 1500 keypoints are detected per image. The computation times correspond to the description and matching of all keypoints.

A. Alahi, R. Ortiz, and P. Vanderghenst. [FREAK: Fast Retina Keypoint](#). In *IEEE Conference on Computer Vision and Pattern Recognition*,

# Binary descriptors

- Gil Levi's CV blog:
  - <https://gilscvblog.wordpress.com/>