MLRF Lecture 03

J. Chazalon, LRDE/EPITA, 2019

Local feature descriptors

Lecture 03 part 03

Introduction

Local feature **detectors** give us the same feature under several perturbations: perspective, illumination, blur…

Local feature descriptors will associate a **vector to each local feature**.

Such description vector should be:

- **Compact** – to enable fast indexing and matching
- **Discriminative** – to enable object recognition
- **Robust to perturbations** – to tolerate real conditions

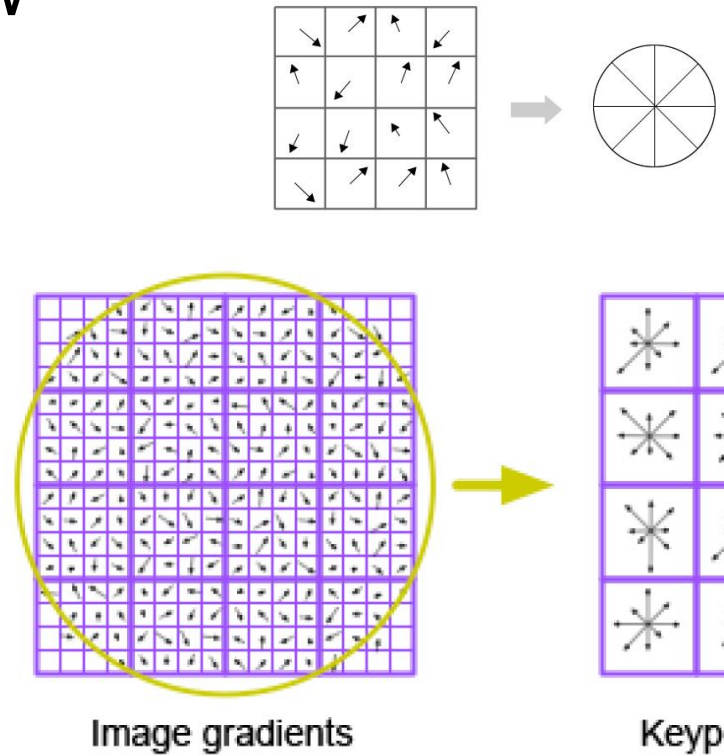
We will focus on two widely used descriptors for their pedagogical interest:

- **HoG** (Histogram of gradients), used in SIFT
- **BRIEF** (Binary Robust Independent Elementary Features), used in ORB

HoG
(Histogram of Gradients)

HoG: Algorithm overview

1. (optional) global image normalisation
2. compute the gradient image in x and y
3. compute gradient histograms
4. normalise across blocks
5. flatten into a feature vector
6. (quantify to integers)

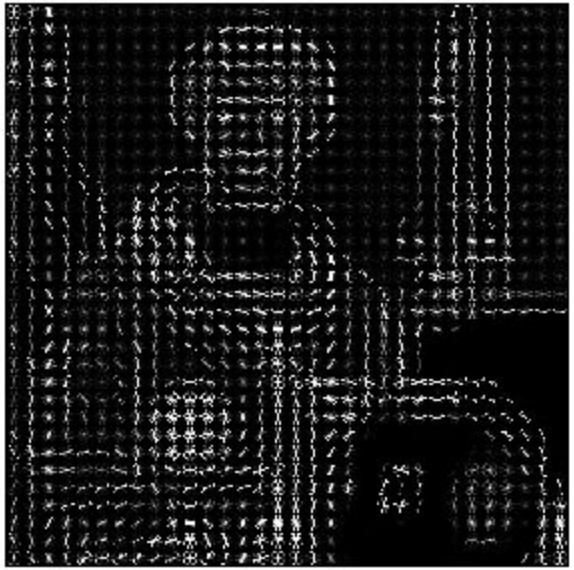


Dalal, N. and Triggs, B., “Histograms of Oriented Gradients for Human Detection,” IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005, San Diego, CA, USA.
David G. Lowe, “Distinctive image features from scale-invariant keypoints,” International Journal of Computer Vision, 60, 2 (2004), pp. 91-110.

HoG: Example

Input image

Histogram of Oriented Gradients



https://scikit-image.org/docs/dev/auto_examples/features_detection/plot_hog.html

HoG summary

Pros

Very stable over illumination changes, perspective changes, blur

Cons

Slow to compute.

Quite large (128 bytes for original SIFT)

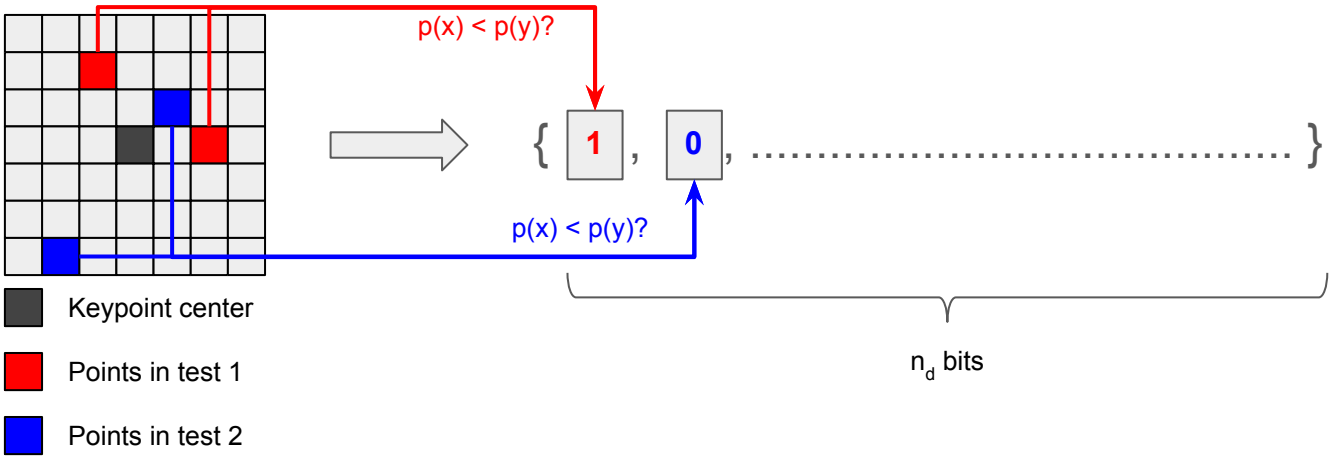
BRIEF
(Binary Robust Independent Elementary Features)

BRIEF

Very spiky otherwise, like derivatives.

General idea:

1. Sample pairs of points {p(x), p(y)} in the **smoothed** keypoint neighborhood
2. Compute a simple **binary test**: p(x) < p(x)
3. Accumulate the results of n_d tests to form a **binary vector of n_d bits** (256 in ref.)



M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “Brief: Binary robust independent elementary features,” in European conference on computer vision, 2010, pp. 778–792.

BRIEF: sampling strategies

G I: $(x_i,y_i) \sim$ i.i.d. Uniform(-S/2,+S/2)

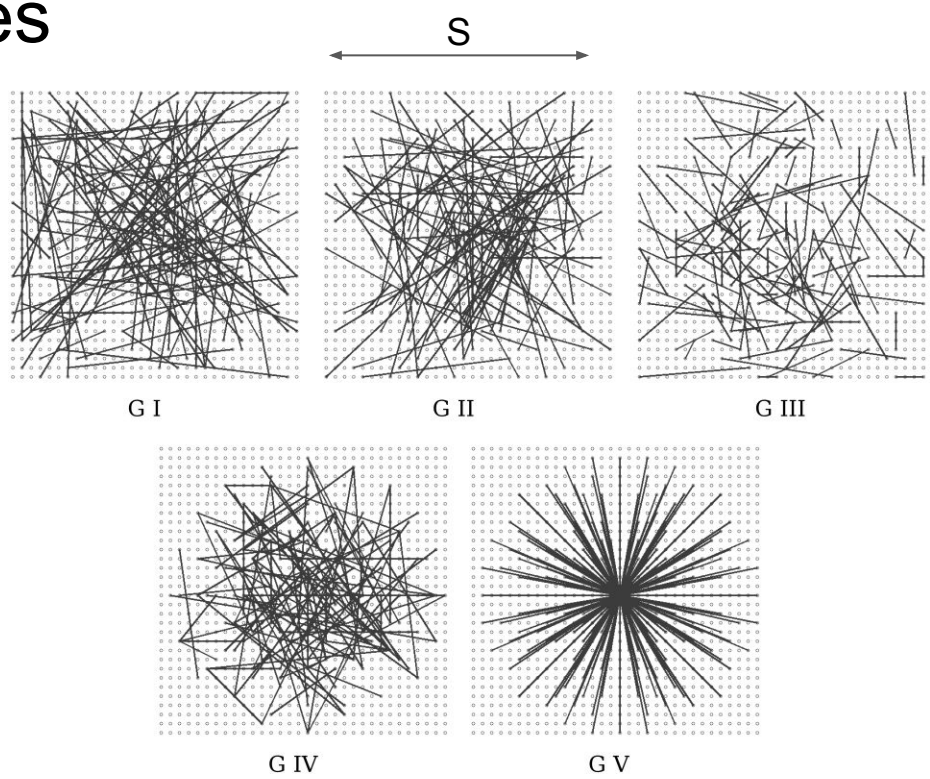
G II: $(x_i,y_i) \sim$ i.i.d. Gaussian(0, $^1/_{25}$ S²)

G III: $x_i \sim$ i.i.d. Gaussian(0, $^1/_{25}$ S²)
$y_i \sim$ i.i.d. Gaussian(x_i, $^1/_{100}$ S²)

G IV: (x_i,y_i) randomly sampled from discrete locations of a coarse polar grid introducing a spatial quantization.

G V: $x_i = (0,0)$
y_i takes all possible values on a coarse polar grid containing n_d points.

What is the best approach?

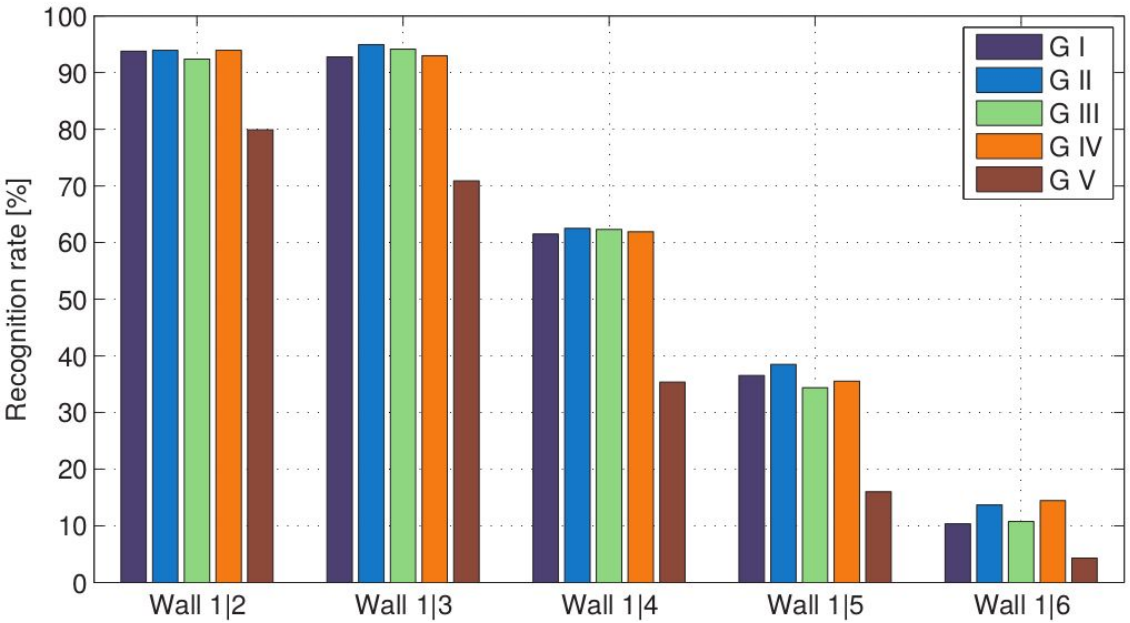


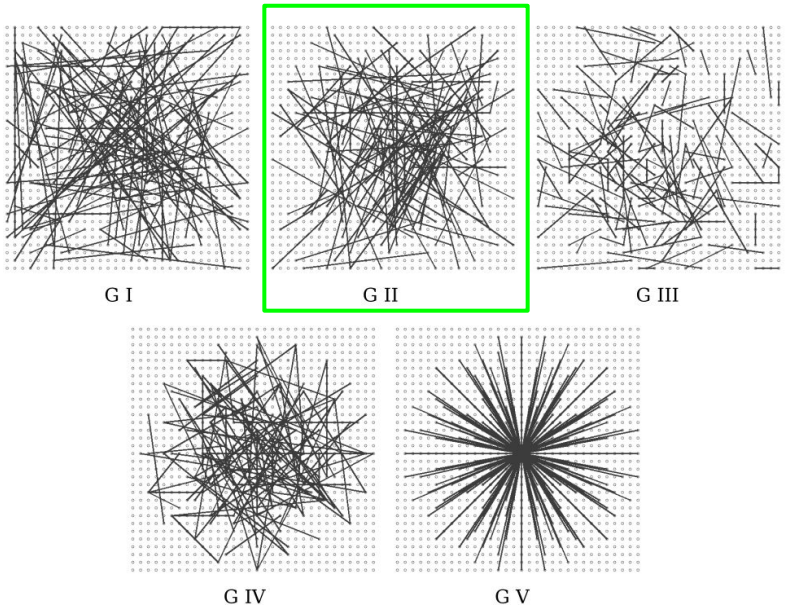
M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “Brief: Binary robust independent elementary features,” in European conference on computer vision, 2010, pp. 778–792.

BRIEF: sampling strategies

G II: $(x_i, y_i) \sim$ i.i.d. Gaussian$(0, {}^{1}/_{25}\, S^2)$

Random sampling around the center.





M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “Brief: Binary robust independent elementary features,” in European conference on computer vision, 2010, pp. 778–792.

BRIEF summary

Pros

Very fast to compute

Very fast to match
(SSE acceleration for Hamming distance)

Very compact to store

Cons

Less robust than HoG (SIFT), DoH (SURF) on several real cases.

Invariance checks

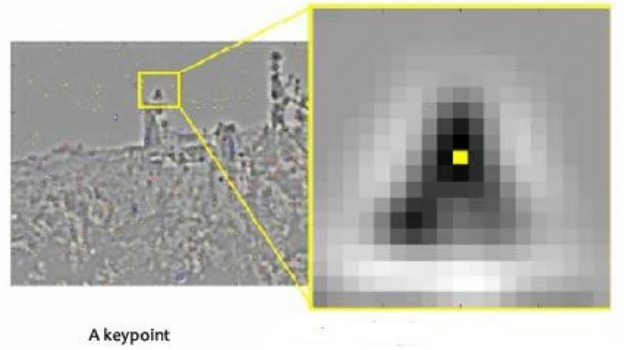
Rotation invariance

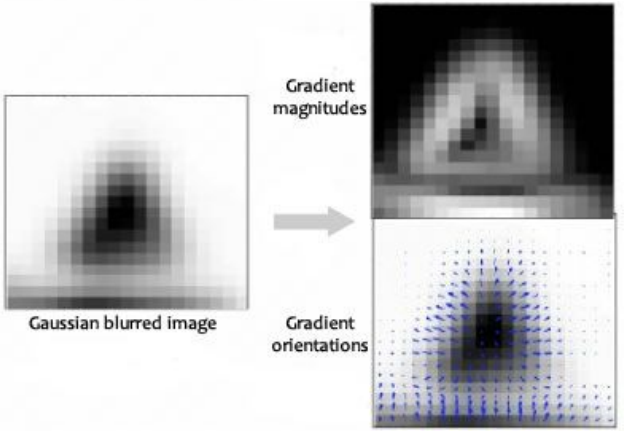
Add an angle measure!

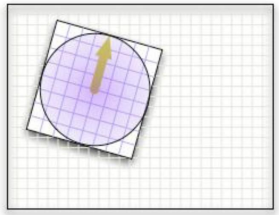
Take the main gradient orientation.
(Take the average around the keypoint.)

⇒ We now have for each keypoint:
- Coordinates
- **Orientation**

⇒ Descriptor: computed over **normalized patch**







Main patch orientation

http://aishack.in/tutorials/sift-scale-invariant-feature-transform-keypoint-orientation/

Scale invariance

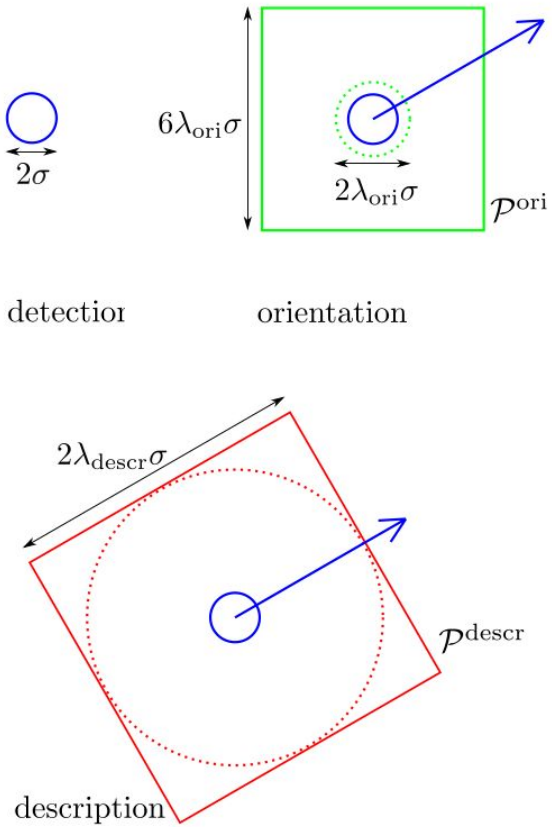
Multi scale feature detection and computation:

- Add a keypoint for each relevant scale
- Possibly several keypoints at the same position!

⇒ We now have for each keypoint:

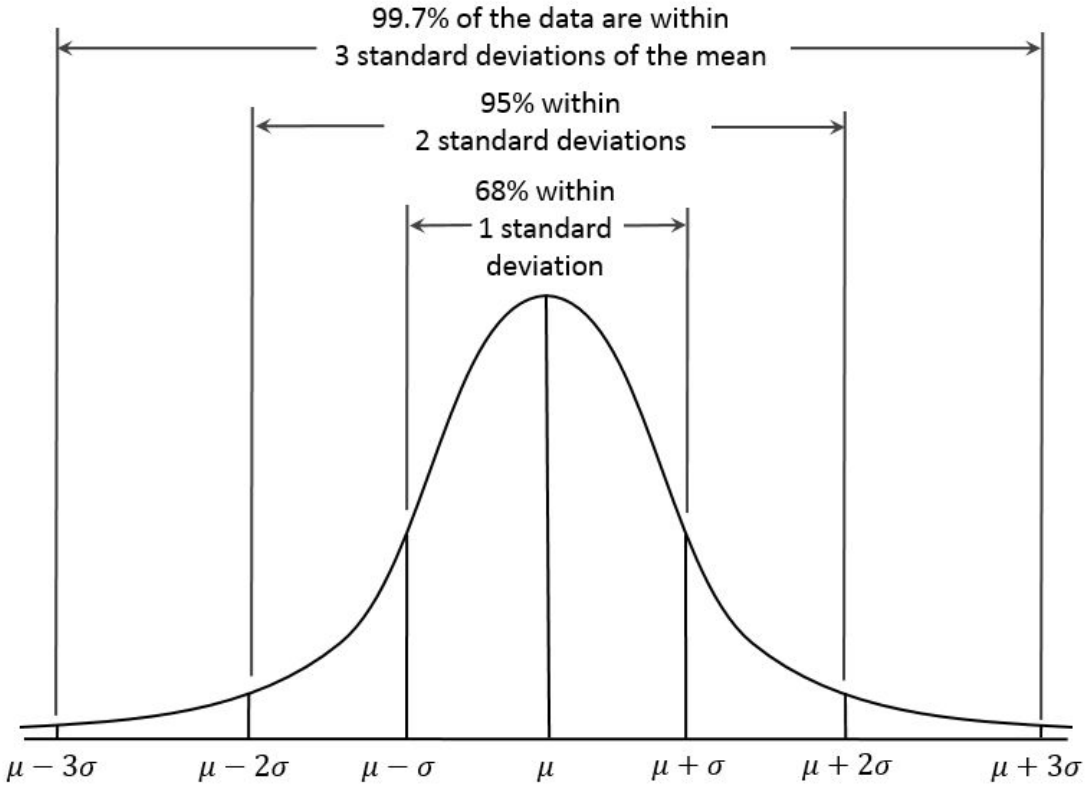
- Coordinates
- Orientation
- **Scale**

⇒ Descriptor: computed on a **scaled patch**



I. Rey Otero and M. Delbracio, “Anatomy of the SIFT Method,” Image Processing On Line, vol. 4, pp. 370–396, Dec. 2014.

(Reminder: Gaussian's sigma vs window size)

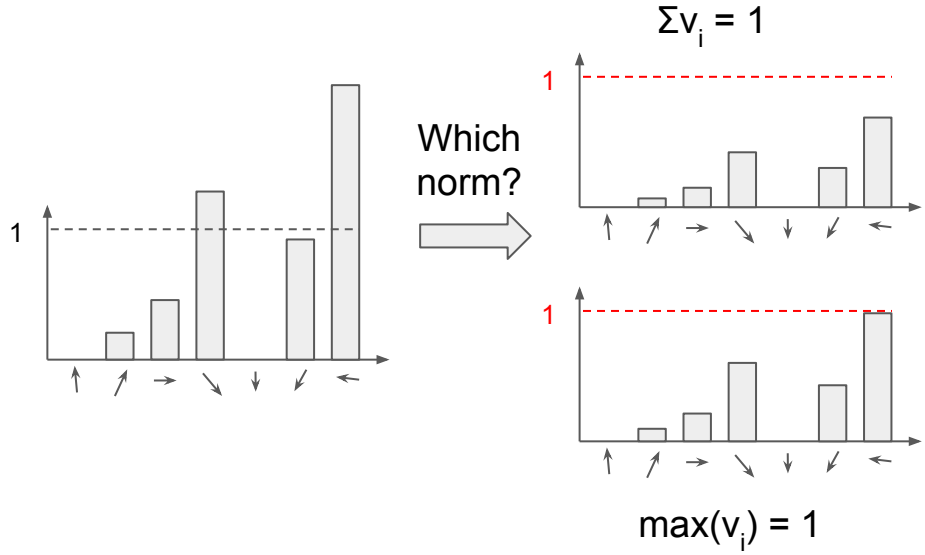


Dan Kernler, https://en.wikipedia.org/wiki/File:Empirical_Rule.PNG

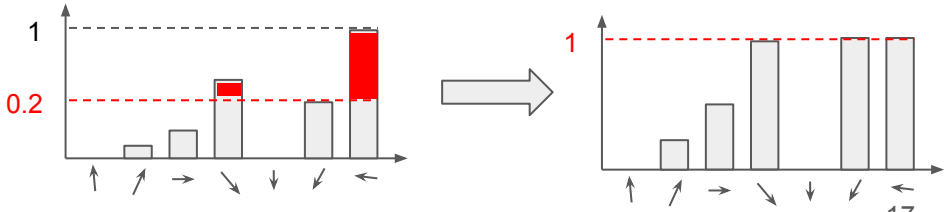
Illumination invariance

SIFT approach:

1. normalize the vector
   ⇒ Solves Affine but what non-linear sources like camera saturation?



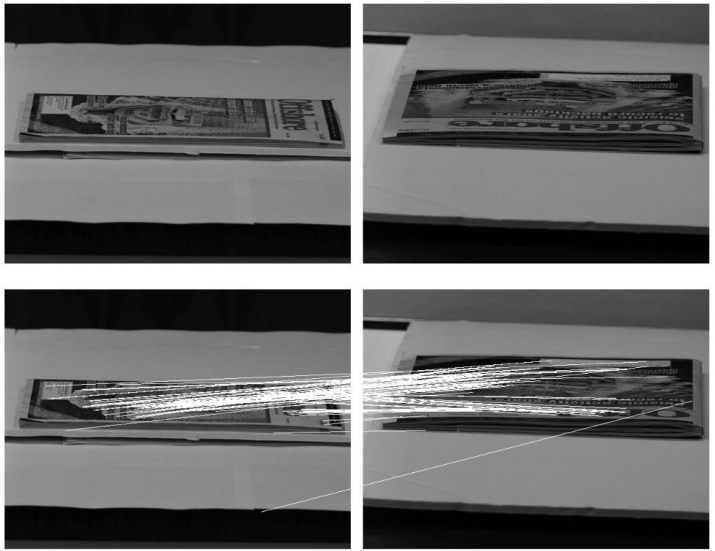
2. Cap the vector elements to 20% (!) and renormalize
   ⇒ Now we have some illumination invariance



Viewpoint invariance

Well, mostly…

Better, but more complex approaches can tolerate extreme viewpoint change.



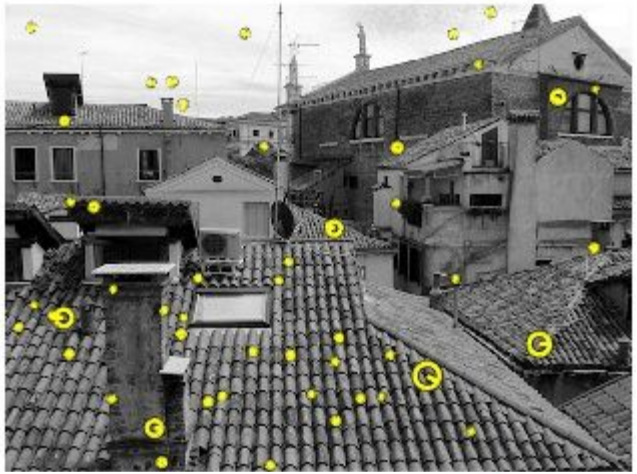
See, for instance:

J.-M. Morel and G. Yu, “ASIFT: A New Framework for Fully Affine Invariant Image Comparison,” SIAM Journal on Imaging Sciences, vol. 2, no. 2, pp. 438–469, Jan. 2009.

Complete pipelines

SIFT (Scale invariant feature tr.)

1. Construct Scale Space
2. Take Difference of Gaussians
3. Locate DoG Extrema
4. Sub Pixel Locate Potential Feature Points
5. Filter Edge and Low Contrast Responses
6. Assign Keypoints Orientations
7. Build Keypoint Descriptors
8. Matching, etc.



D. G. Lowe, “Object recognition from local scale-invariant features,” in Computer vision, 1999. The proceedings of the seventh IEEE international conference on, 1999, vol. 2, pp. 1150–1157
Illustrations: http://www.vlfeat.org/overview/sift.html

ORB (oriented FAST and rotated BRIEF)

1. Use FAST in pyramids to detect stable keypoints
2. Select the strongest features using FAST or Harris response
3. Finds their orientation using first-order moments
4. Computes the descriptors using BRIEF
   where the coordinates of random point pairs
   are rotated according to the measured orientation

E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: an efficient alternative to SIFT or SURF. In Computer Vision (ICCV), 2011 IEEE International Conference on, pages 2564–2571. IEEE, 2011.

Conclusion about feature extraction

Selection of appropriate features:

- It is a **critical** decision
- Depends on the **specific** application

Features must:

- Be **invariant** to data variations (depending on the application):
  rotation, perspective, noise, blur, shape distortion, illumination, compression…
- Have **low dimensionality** for fast training, matching, reasonable storage

Features determine the **type of information** to work with:

- gray-level, binary, color image,
- contours, vectorization, skeleton

Features also determine the **type of classifier / indexer**