
Computer vision

Computer Vision

Lecture 3: Features II

Dr. Dina Khattab

dina.khattab@cis.asu.edu.eg

Scientific Computing Department

Instructor:	Dr. Dina Khattab
Email:	dina.khattab@cis.asu.edu.eg
Office:	Main Building – 4th floor – Room 302
Office Hours:	Monday 12:00 - 2:00 PM Thursday 11:00 AM to 12:00 PM

Agenda

- Feature Descriptors
  - SIFT
  - HOG

SIFT: SCALE INVARIANT FEATURE TRANSFORM

SIFT: Scale Invariant Feature Transform

For better image matching, Lowe's goals were:

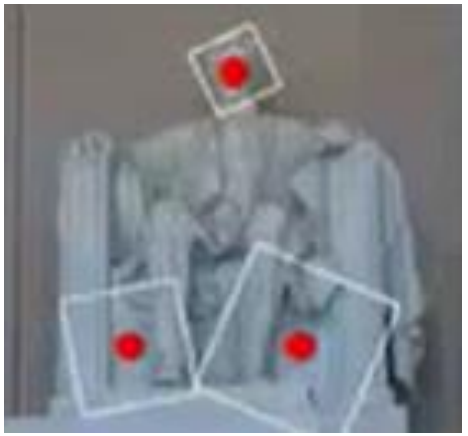
- To develop an interest operator – a *detector* – that is invariant to scale and rotation.
- Also create a *descriptor* that was robust to the variations corresponding to typical viewing conditions.

D.Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". IJCV 2004

How to extract SIFT

Test Image

Detector: Where are the Local features?



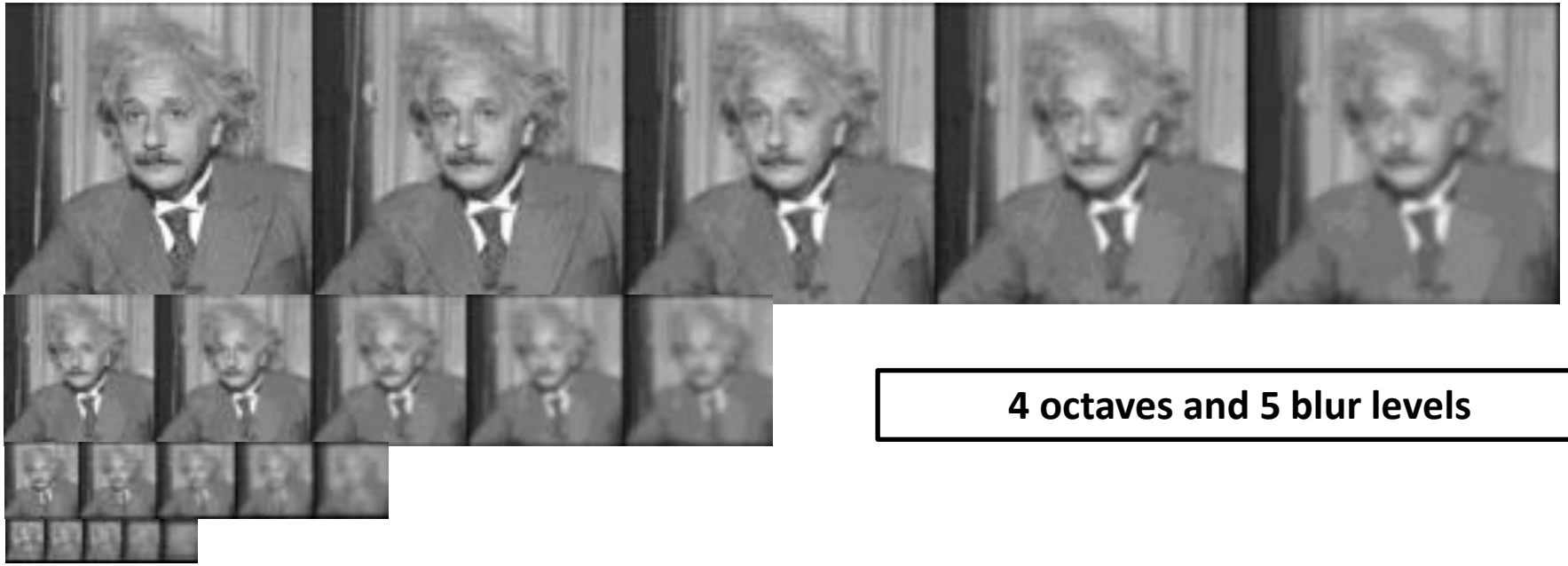
Descriptor: how to describe them?

SIFT Algorithm Steps

- Step 1: Constructing a scale space.
- Step 2: Laplacian of Gaussian approximation.
- Step 3: Keypoint localization.
- Step 4: Eliminate edges and low contrast regions.
- Step 5: Orientation Assignment.
- Step 6: Keypoint description.

Step 1: Constructing a scale space

- Take the original image and generate progressively blurred out versions using Gaussian filter.
- This is applied in different scale space, that resize the original image into different scales and generate blurred out images in each level.



4 octaves and 5 blur levels

Step 2: Laplacian of Gaussian approximation (LoG)

- Apply LoG (second order derivative) on each blurred image in all scale levels to get keypoints of high contrast (edges and corners), however this is computationally expensive.

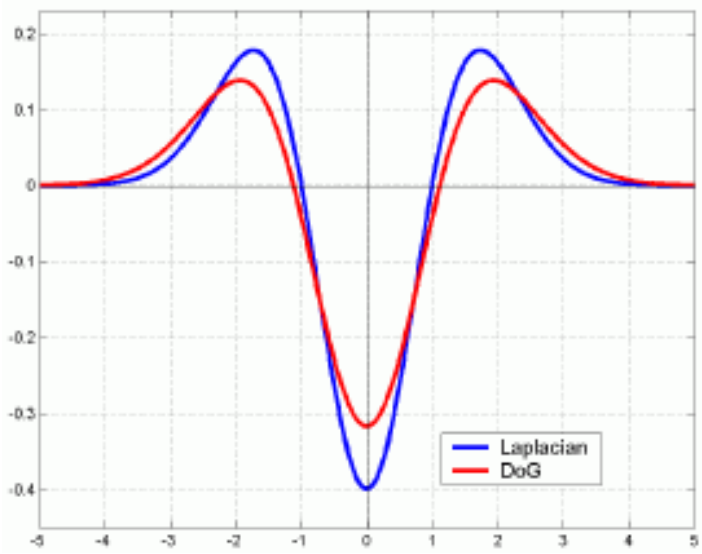
0	1	0
1	-4	1
0	1	0

2D Laplace filter

circularly symmetric - invariant to rotation and scale

Step 2: Laplacian of Gaussian approximation

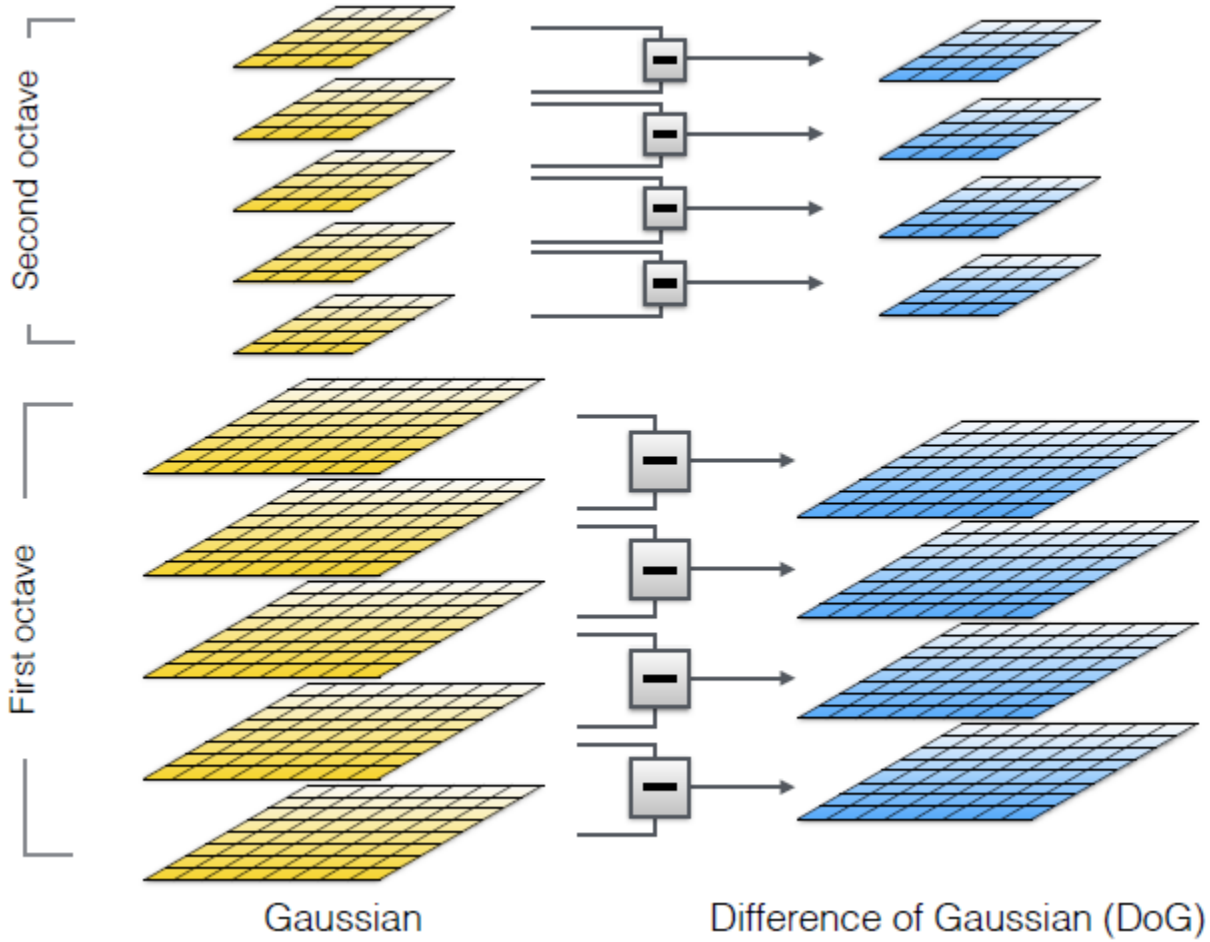
- In practice, the Laplacian is approximated using a Difference of Gaussian (DoG)
- *DoG = G(x, y, Kσ) – G(x, y, σ)*
  (Difference of Gaussians)

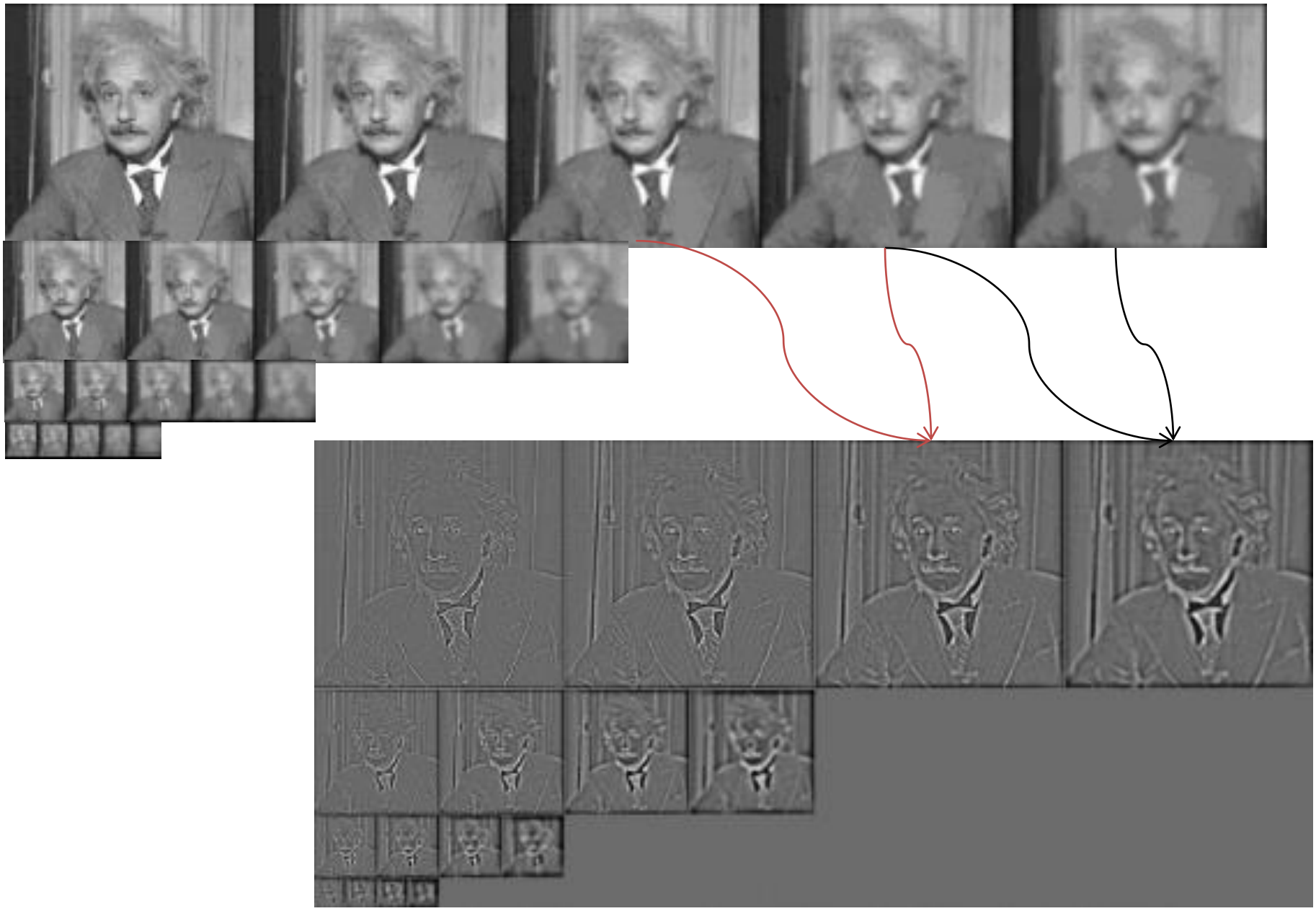


G1 - G2 = DoG

K. Grauman, B. Leibe

- DoG is approximate equivalent to LoG that is less computational (Simple subtraction) that is fast and efficient

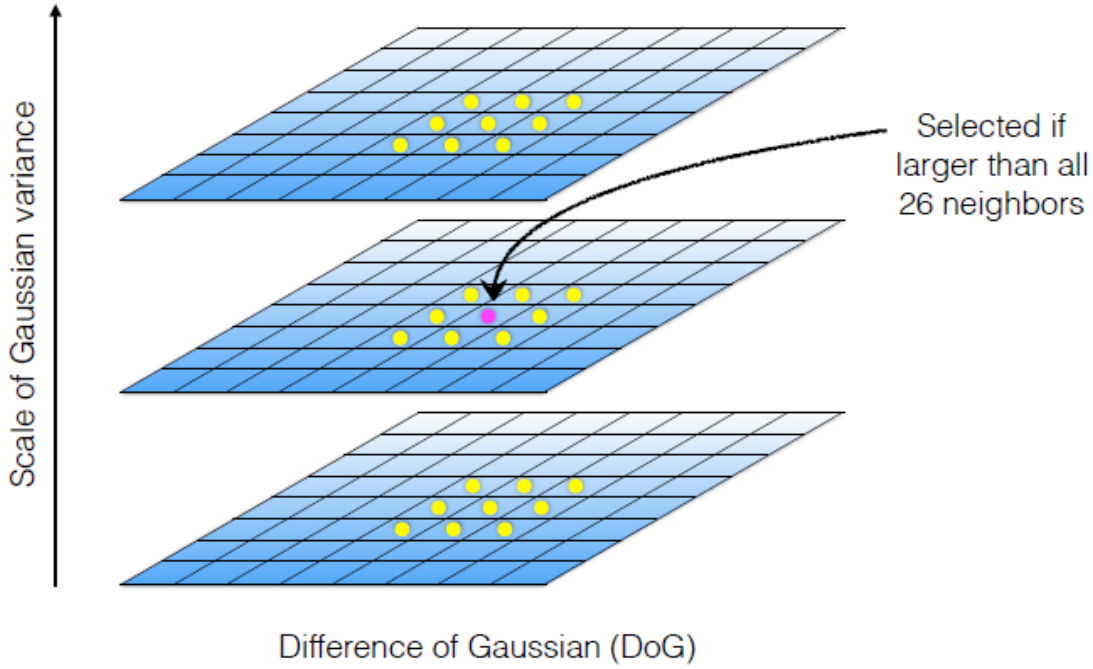




Step 3: Keypoint localization

- Iterate through each point and compare to its 8 neighbors in the current image and 9 neighbors each in the scales above and below.
- A point is marked as key point if it is greater or smaller than all 26 neighbors.

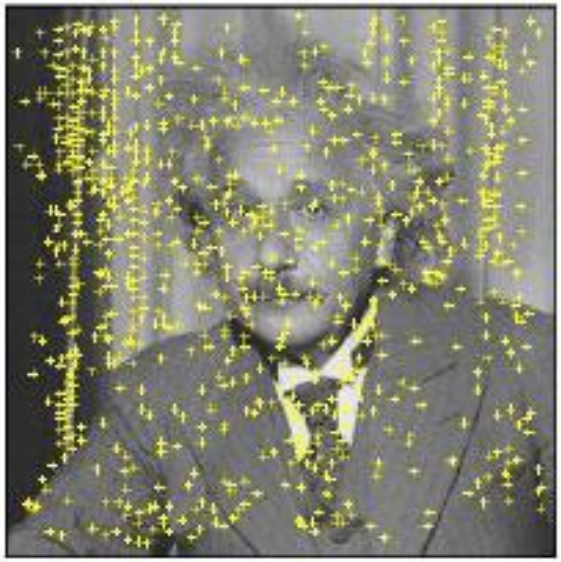
For each max or min found, output is the **location** and the **scale**.



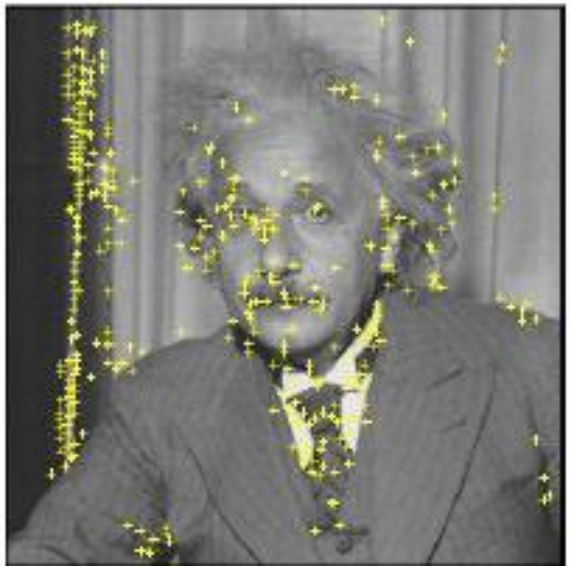
Step 4: Eliminate edges and low contrast regions

- Keypoints generated from last step are too many, some of them lie on edges or don't have enough contrast. In both cases they are not useful and should get rid of them

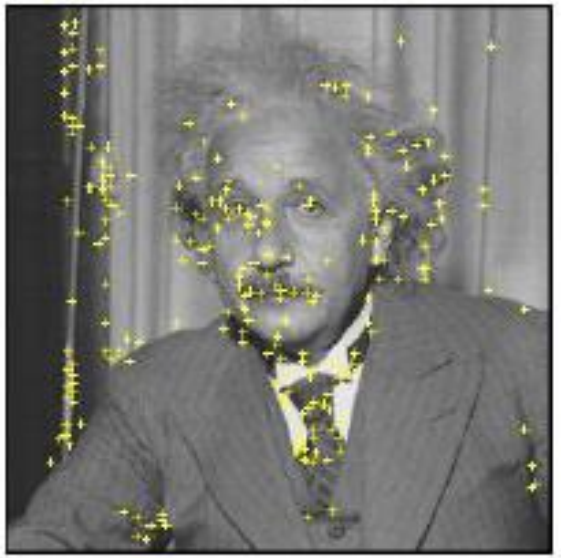
- Reject points with bad contrast:
  - DoG smaller than 0.03 (image value [0,1])
- Reject edges
  - Use Harris detector and keep only corners.



Extrema points



Contrast > C



Not on edge

233x189

832

initial keypoints

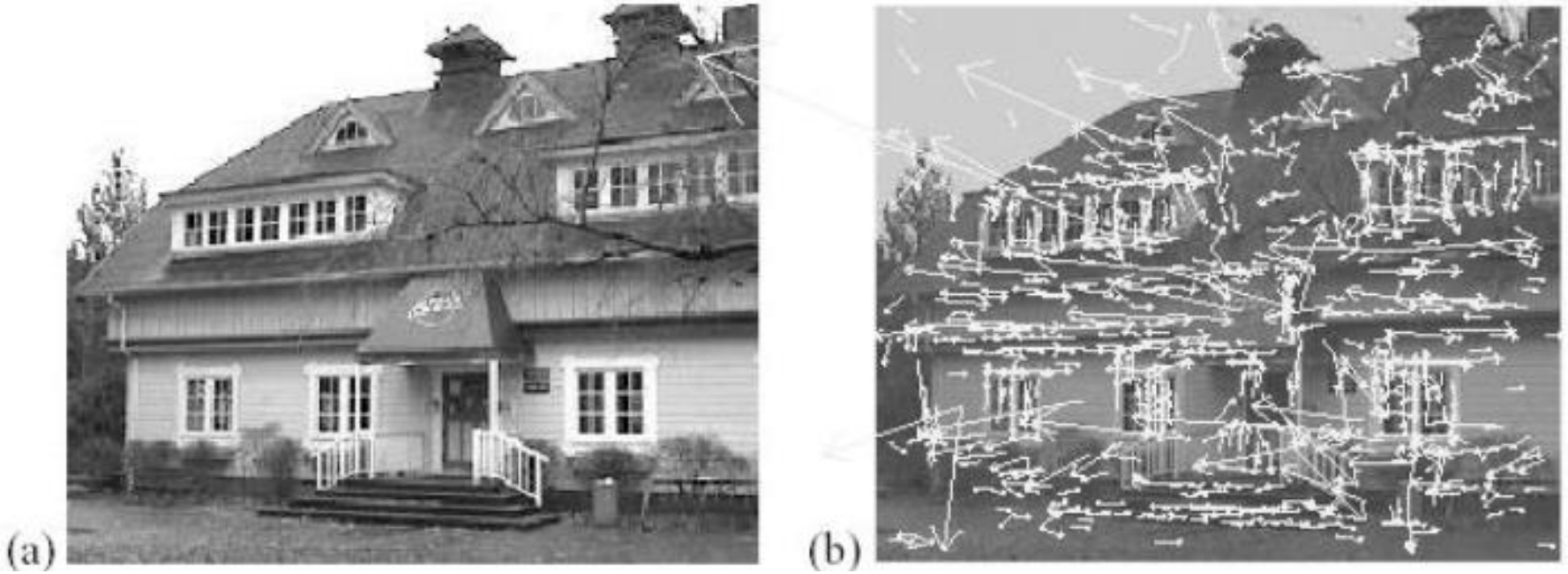
729

keypoints after
gradient threshold

536

keypoints after
Edge removing

Step 5: Orientation Assignment



Step 5: Orientation Assignment

- Take a window (16x16) around each keypoint and calculate the gradient magnitude and orientation of each pixel and assign the most dominant orientation(s) in this region to the keypoint.

  - The gradient of an image:

$$\nabla f = [\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}]$$

  - The gradient direction:

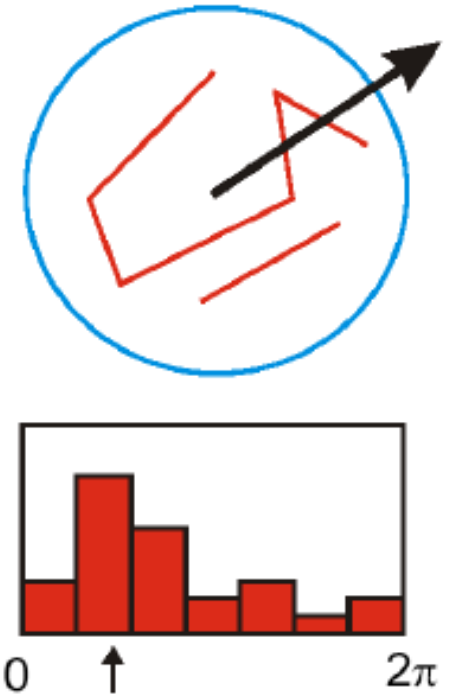
$$\theta = \tan^{-1}(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x})$$

  - The gradient magnitude:

$$\|\nabla f\| = \sqrt{(\frac{\partial f}{\partial x})^2 + (\frac{\partial f}{\partial y})^2}$$

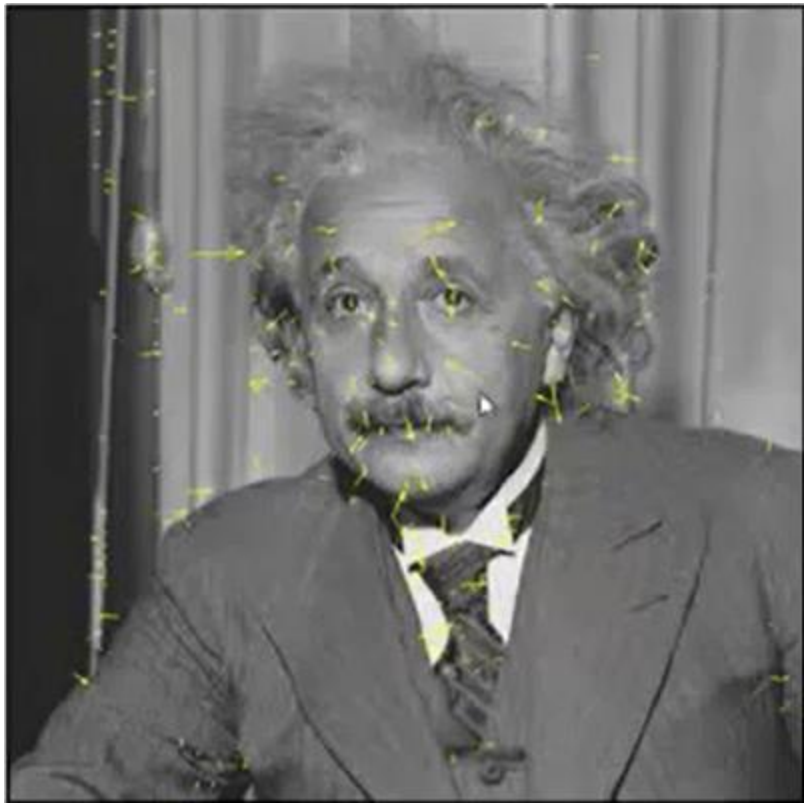
Step 5: Orientation Assignment

- Create histogram of local gradient directions at selected scale – 36 bins (each 10 degrees).
- The keypoint is assigned to the orientation of the highest peak.



Step 6: SIFT Descriptor

- Next is to compute a descriptor for the local image region about each keypoint that is:
  - Highly ***distinctive***
  - As ***invariant*** as possible to variations such as changes in viewpoint and illumination.

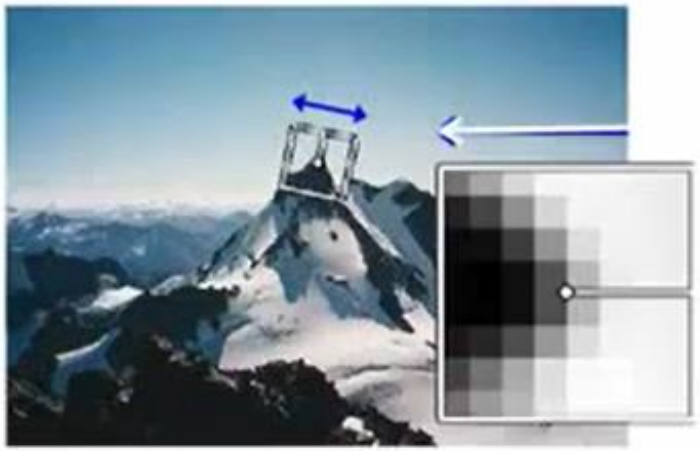


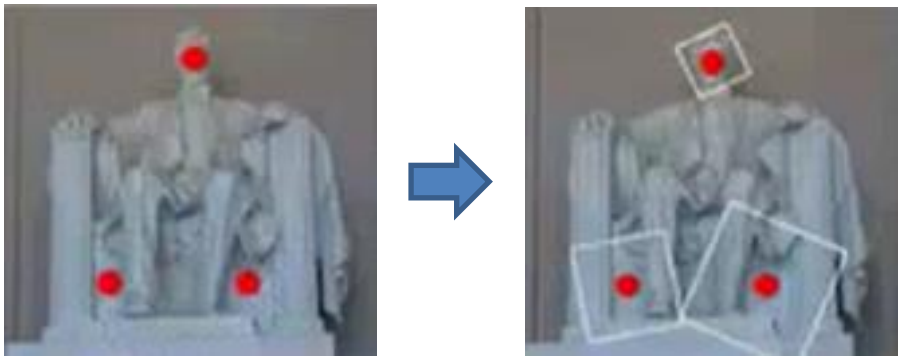
- Each *keypoint* now specifies:
  - location (x, y)
  - Scale σ
  - Gradient magnitude and orientation m, θ

But first… normalization…

(rotation and scale invariant)

- Rotate the window to standard orientation (negative keypoint orientation).
- Scale the window size based on the scale at which the point was found.



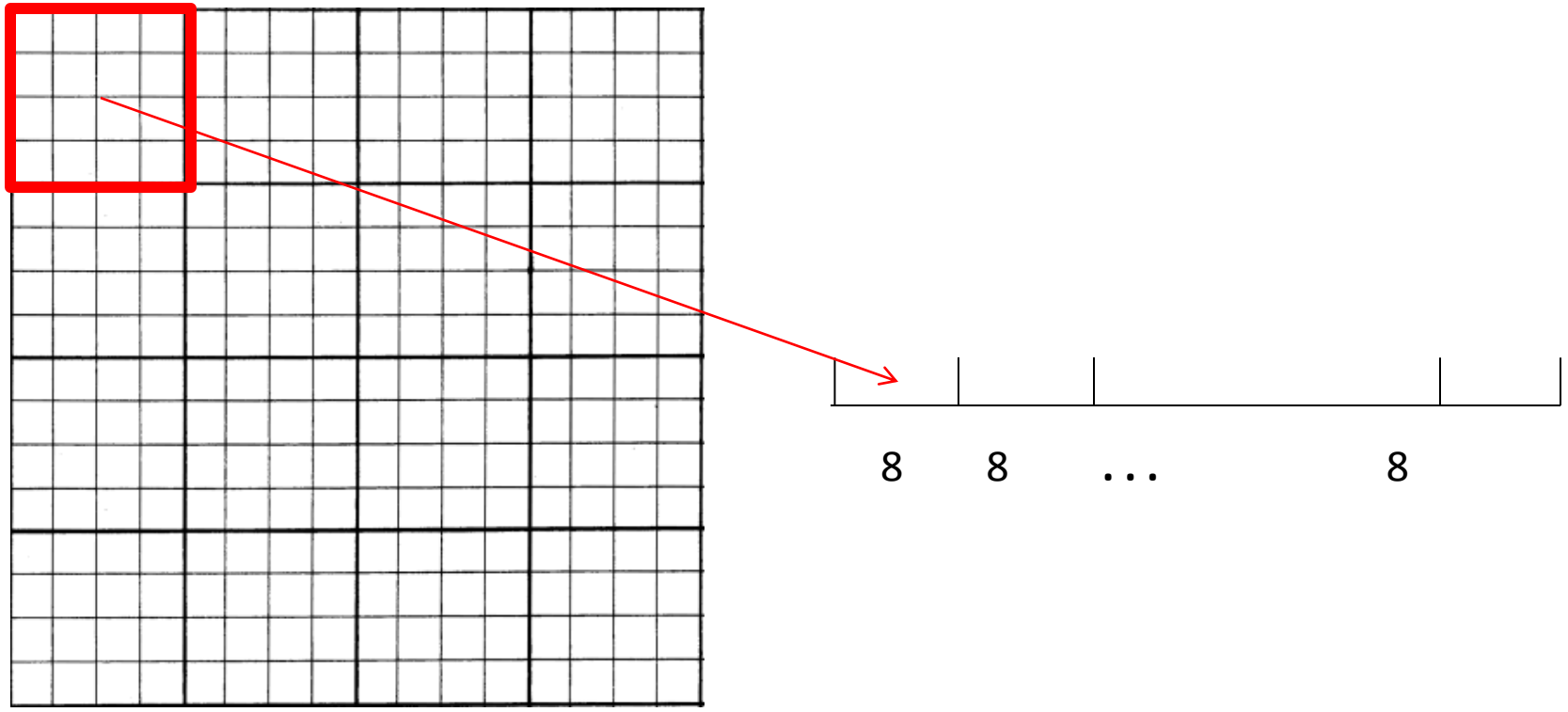


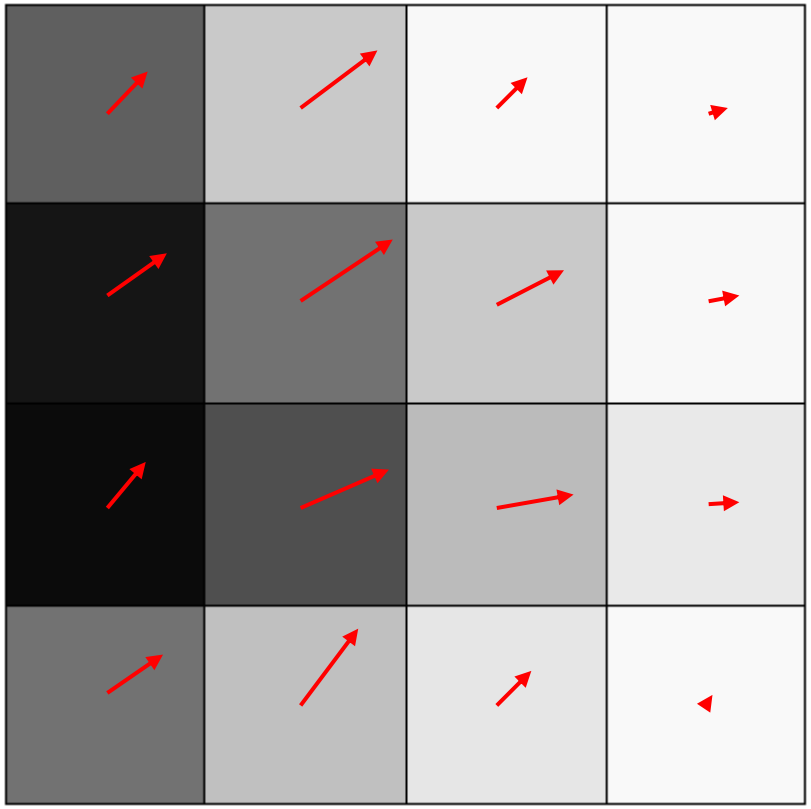
Step 6: SIFT Descriptor

- How can we find corresponding points?

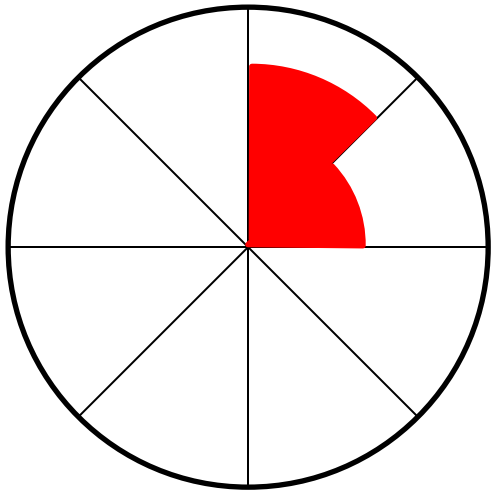
Step 6: SIFT Descriptor

- Take 16x16 window around the keypoint, then broke it into sixteen 4x4 window.
- Calculate gradient magnitude and orientation within each 4x4 window and put into 8-bin histogram (spans 45 degrees)





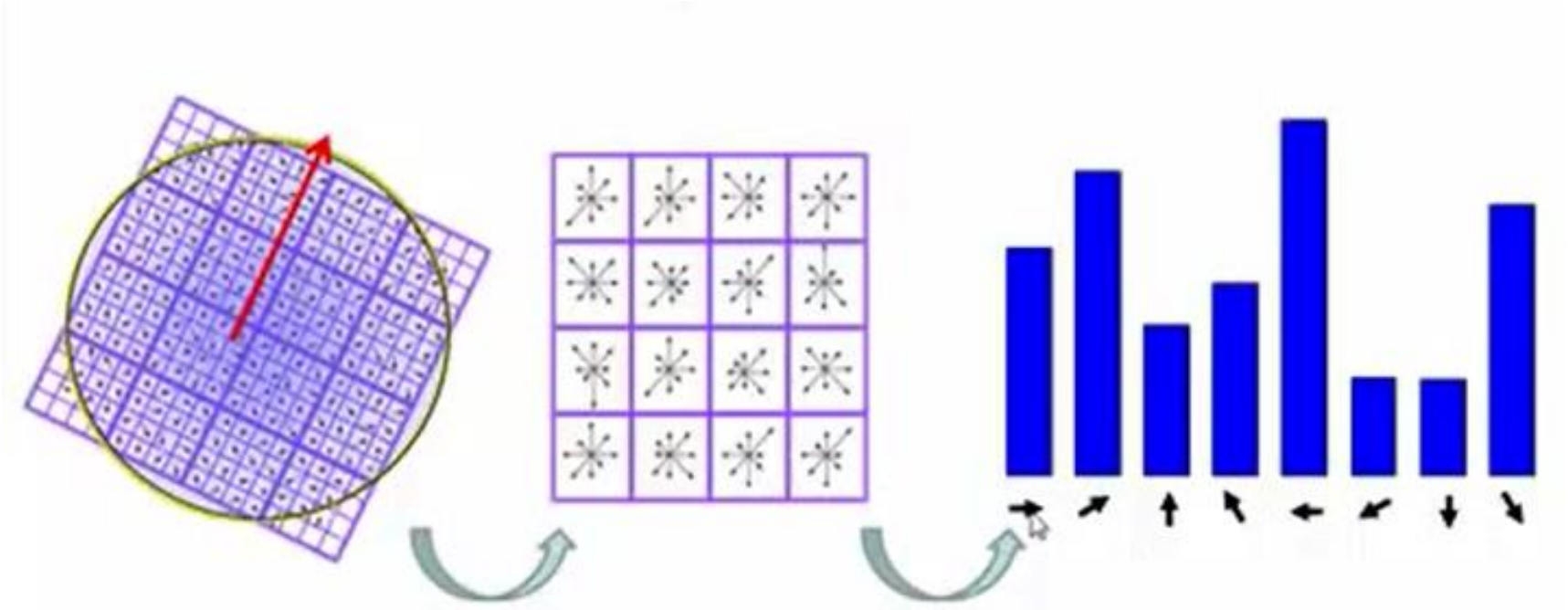
Orientations in each of the 16 pixels of the cell



The orientations all ended up in two bins: 11 in one bin, 5 in the other. (rough count)

Step 6: SIFT Descriptor

- Do this for all 4x4 window so you end up with 4x4x8 = 128 numbers (128 dimensional descriptor).
- The 128 number formalize a “Feature Vector” where the keypoint is uniquely identified.



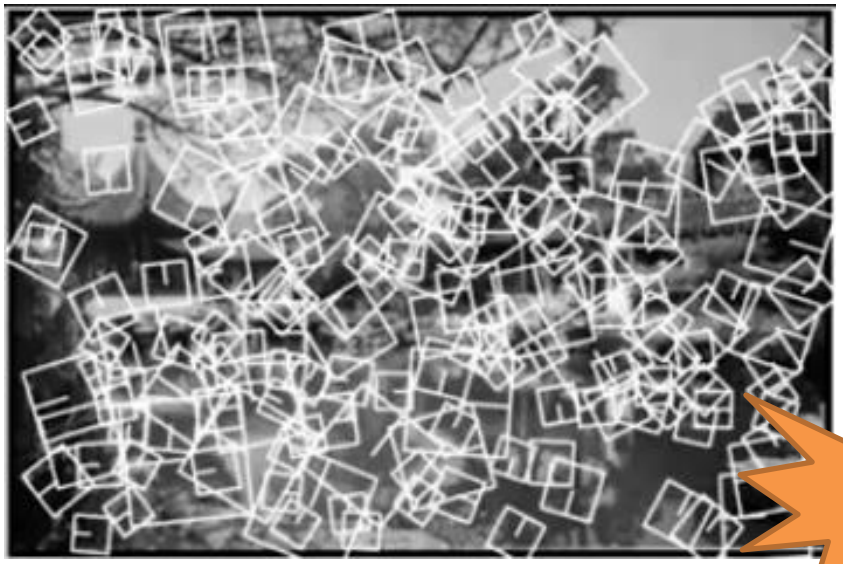
Summary of SIFT Feature

- Descriptor: 128-D
  - 4 by 4 patches each with 8-D gradient angle histogram
  - Normalized to reduce the effect of illumination change.
- Position (x,y)
  - where the feature is located.
- Scale
  - Control the region size for descriptor extraction.
- Orientation
  - to achieve rotation-invariant descriptor.

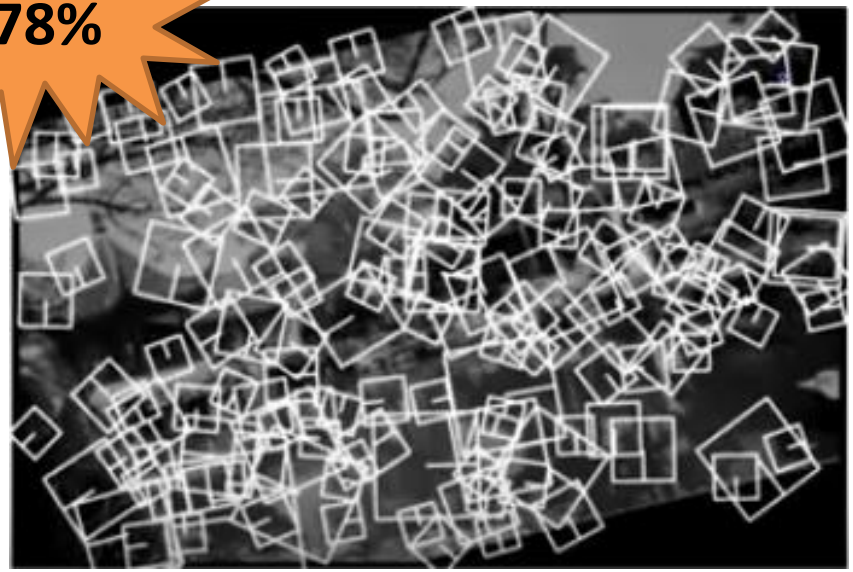
We can compare each vector from image A to each vector from image B to find matching keypoints!

- Euclidean “distance” between descriptor vectors gives a good measure of keypoint similarity

Experimental results



78%



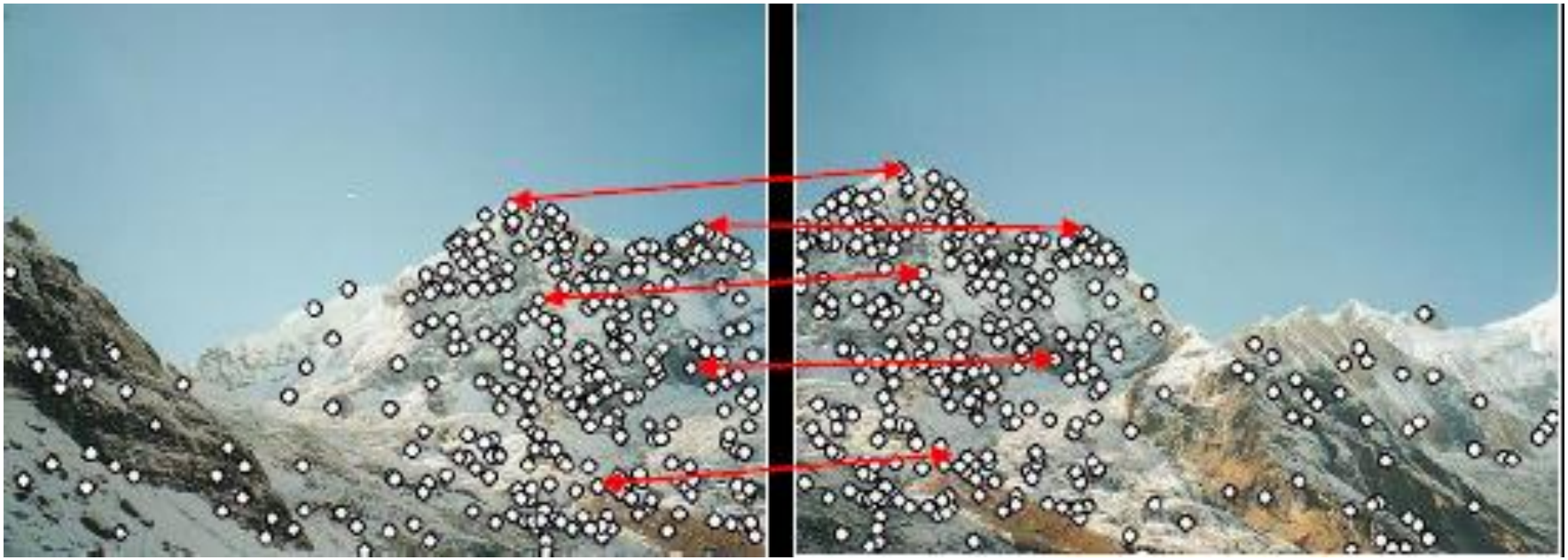
- Keypoints on image after:
  - rotation (15°)
  - scaling (90%)
  - horizontal stretching (110%)
  - change of brightness (-10%) and contrast (90%)
  - addition of pixel noise

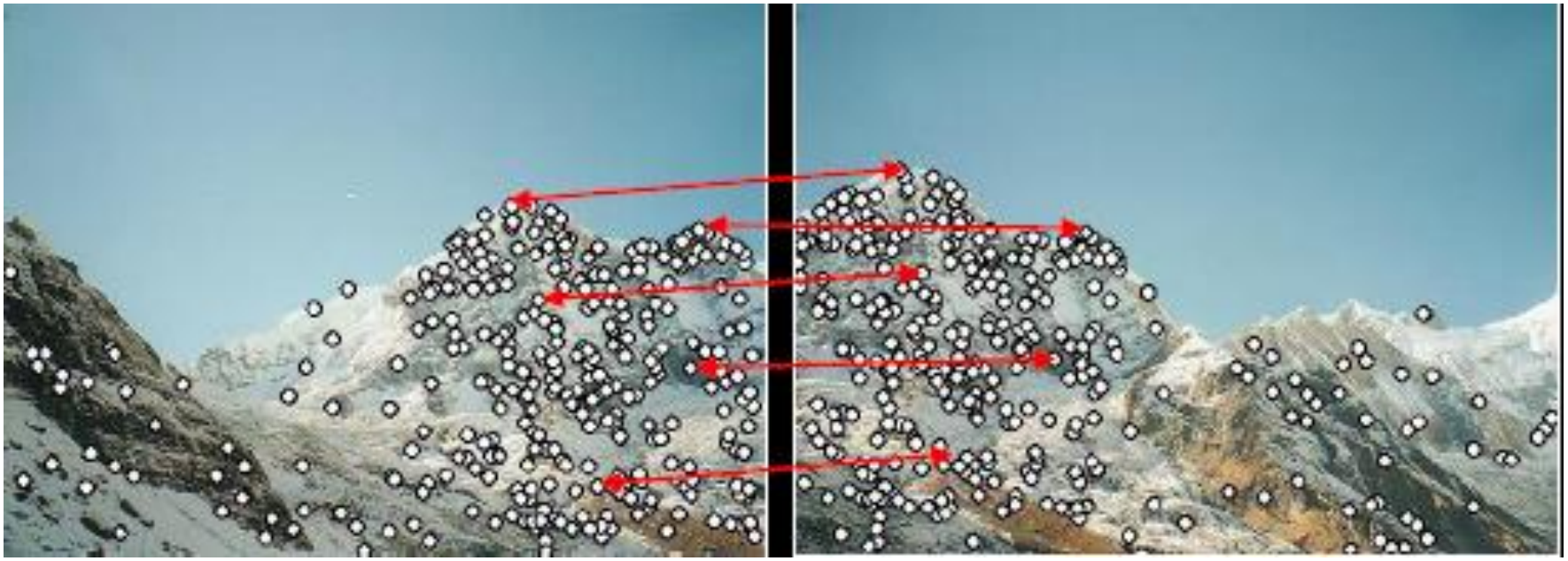
Building Panorama

- Detect and describe in both images (Harris – SIFT - .. etc.).

Building Panorama

- Match features - find corresponding pairs





FEATURE MATCHING

How to match feature points?

We can compare each vector from image A to each vector from image B to find matching keypoints!

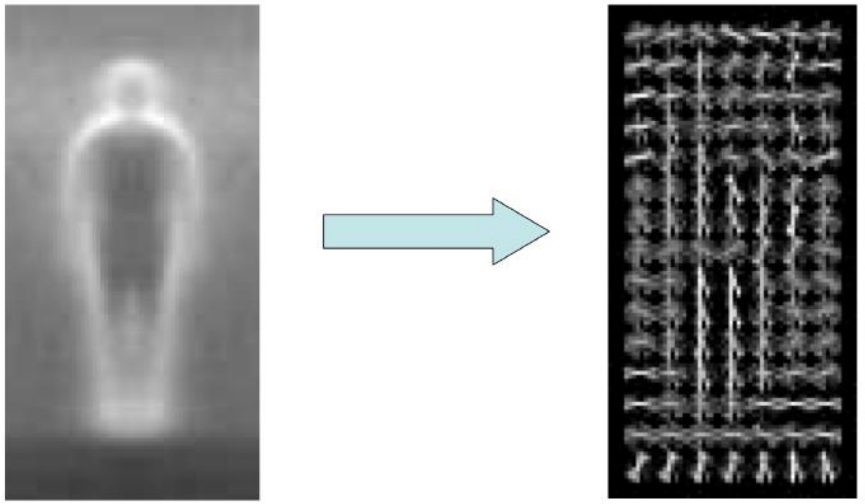
- Euclidean “distance” between descriptor vectors gives a good measure of keypoint similarity

- Could just do nearest-neighbor search.

HISTOGRAM OF ORIENTED GRADIENTS (HOG)

Histogram of Oriented Gradients (HOG)

- The histogram of oriented gradients (HOG) descriptor finds an object within an image that can be discriminated.
- Characterize the local object appearance and shape rather well by the distribution of local intensity gradients or edge directions.

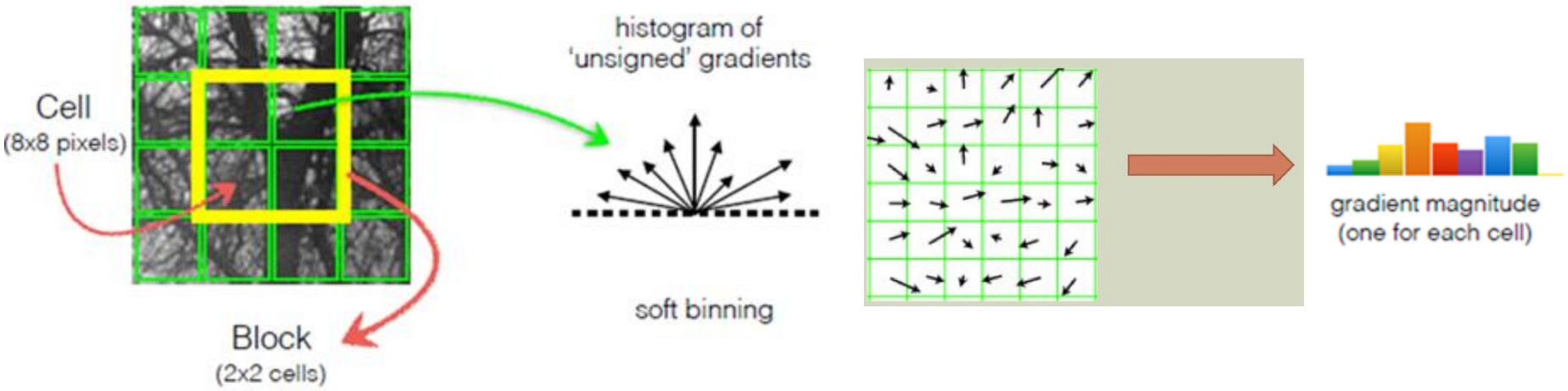


N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In CVPR, pages 886- 893, 2005

Chandrasekhar et al. CHoG: Compressed Histogram of Gradients - A low bit rate feature descriptor, CVPR 2009

HOG algorithm

1. HOG works on images of size 64x128
2. The image is divided into 8×8 pixel size cells and the gradient magnitude and orientation is calculated for each pixel in all cells.
3. A 9-bin histogram is created which corresponds to angels 0 – 180 where it calculates the "unsigned" gradients as a gradient and its negative are represented by the same bin.
4. A bin is selected based on the direction, and the value that goes into the bin is selected based on the magnitude.



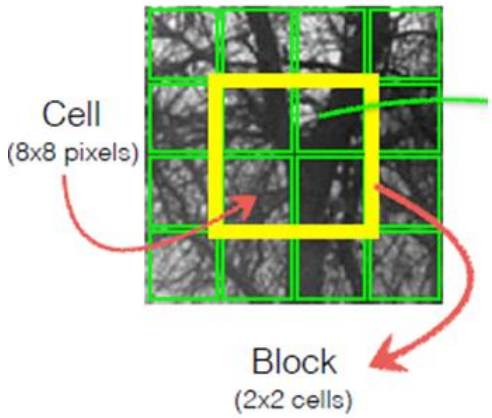
Recap: Image gradients

Gradient magnitude $$\left|\nabla f(x,y)\right| = \sqrt{f_x^2 + f_y^2}$$

Gradient direction $$\theta = \tan^{-1}\frac{f_x}{f_y}$$

HOG algorithm

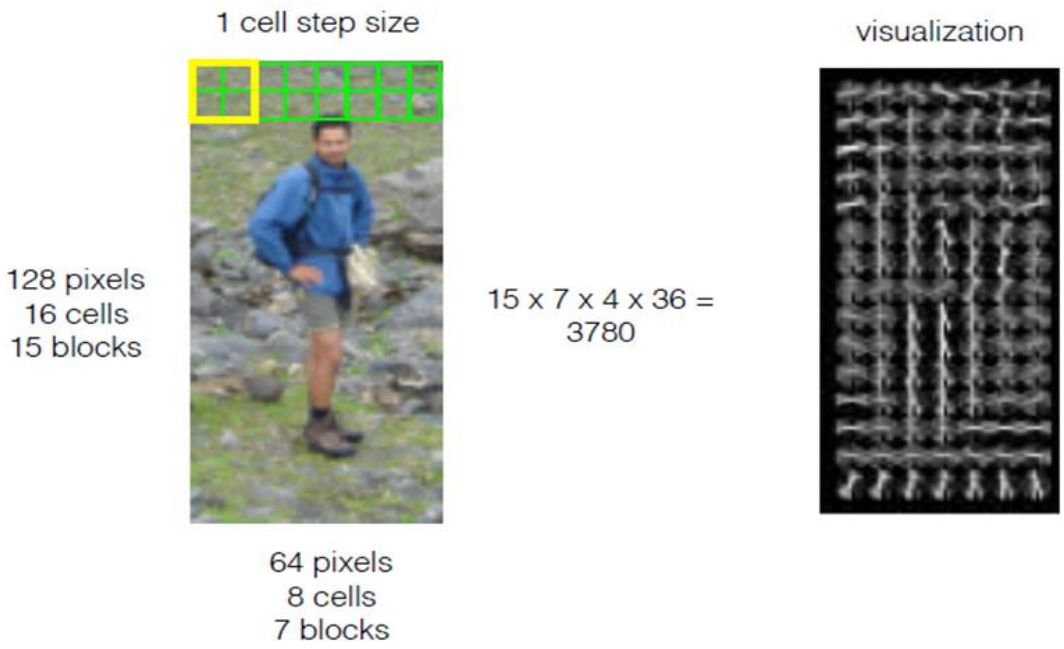
5. The image is divided to a bigger sized blocks of 16x16 pixels (2x2 cells), where each block has 4 histograms that are concatenated into a 1D vector of length 36
6. The window is shifted each time by 8 pixels (1 cell) and the 36 length vector corresponding to each block is normalized to be illumination invariant.



Single scale, no dominant orientation

HOG algorithm

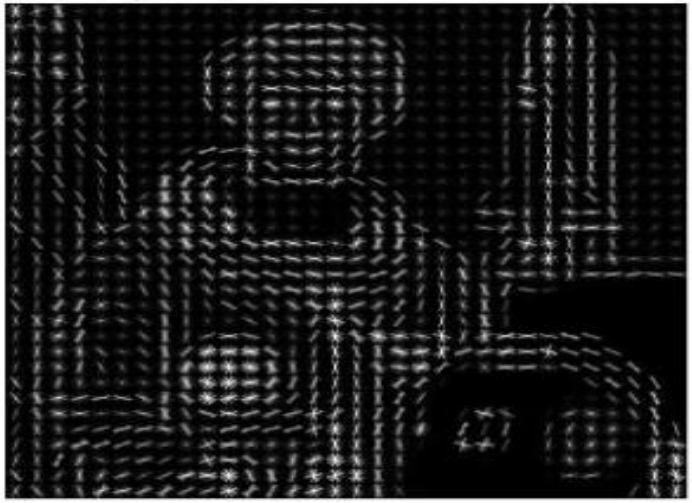
7. The window shift creates a size of 7 horizontal and 15 vertical blocks with total of 105 blocks.
8. All the 36 length normalized vector of the 105 blocks are concatenated in one giant vector of size 36x105=3780 dimensional vector which represents the feature vector of the HOG descriptor.

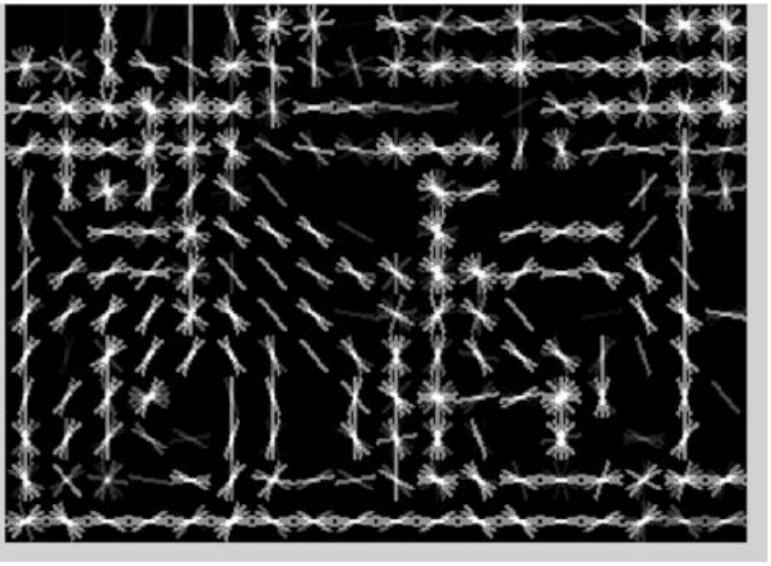


Visualizing HOG

Input image

Histogram of Oriented Gradients





Difference between HOG and SIFT

- HOG is usually used to describe entire images. SIFT is used for key point matching.
- SIFT histrograms are oriented towards the dominant gradient. HOG is not.
- HOG gradients are normalized using neighborhood bins.
- SIFT descriptors use varying scales to compute multiple descriptors.

Credit for

CS 4495 Computer Vision (Spring 2015)

A. Bob - College of Computing, Georgia Tech.

CS131 "Computer Vision: Foundations and Applications" by University of Stanford (Fall 2019)

Elsayed Hemayed – Computer Engineering Dept.

Faculty of Engineering, Cairo University

16-385 Computer Vision

Kris Kitani - Carnegie Mellon University