CAP5415-Computer Vision
Lecture 5 and 6-Finding Features, Affine Invariance, SIFT

Ulas Bagci

bagci@ucf.edu

Outline

- Concept of Scale
  - Pyramids
  - Scale-space approaches briefly
- Scale invariant region selection
- SIFT: an image region descriptor

- *David Lowe, IJCV 2004 paper [Please read it!]*

Reminder: Motivation

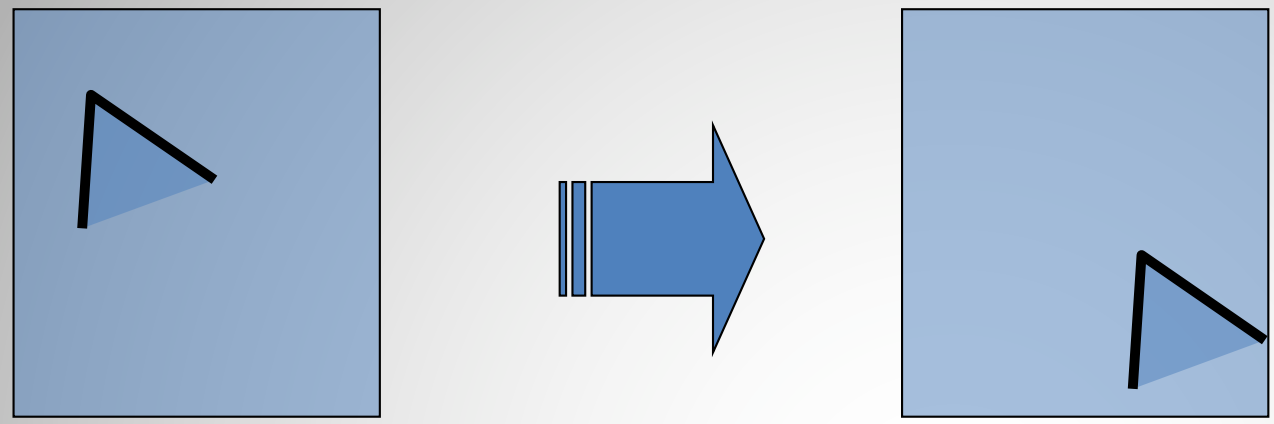
- Image Matching
  - Fundamental aspect of many problems
    - Object Recognition
    - 3D Structures
    - Stereo Correspondence
    - Motion Tracking
    - ….

Reminder: Motivation

- Image Matching
  - Fundamental aspect of many problems
    - Object Recognition
    - 3D Structures
    - Stereo Correspondence
    - Motion Tracking
    - ….

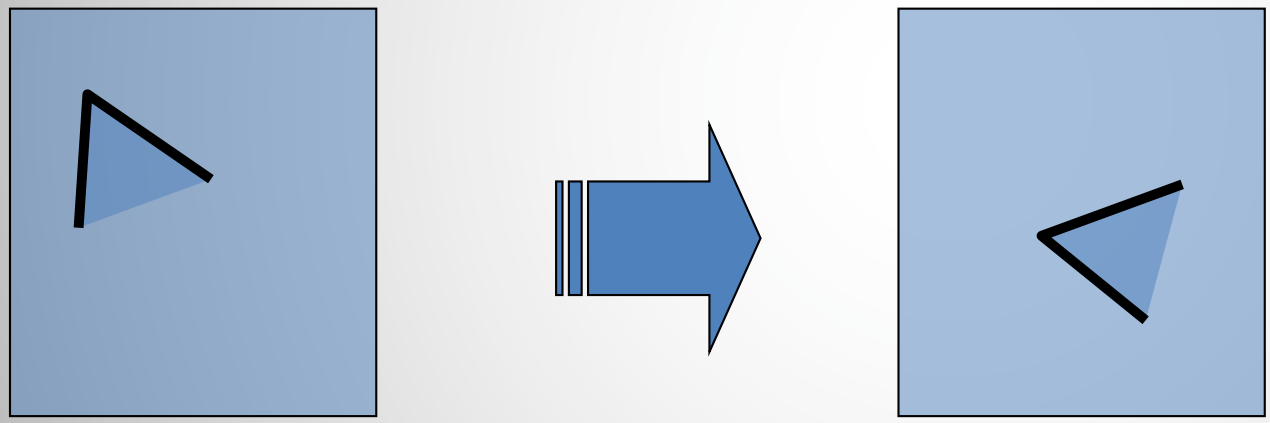
What are the desired features to conduct these tasks?

Review of the Corner Detection

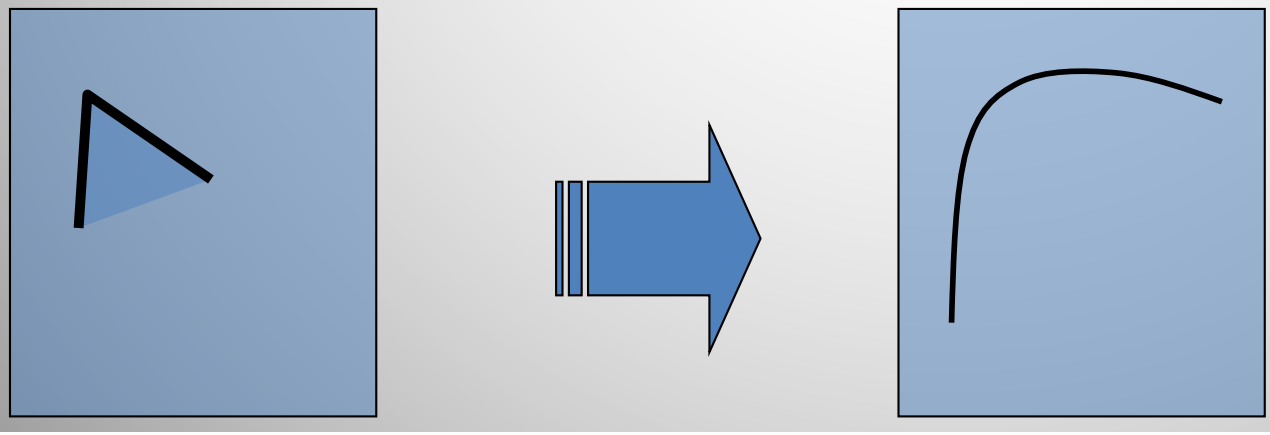


Corners are invariant to

translation ✔



rotation ✔



scaling ✖

SCALE

- The extrema in a signal and its first a few derivatives provide a useful general purpose description for many kinds of signals
  - Edges
  - Corners

SCALE

- The extrema in a signal and its first a few derivatives provide a useful general purpose description for many kinds of signals
  - Edges
  - Corners
- In physics, objects live on a range of scales.

SCALE

- The extrema in a signal and its first a few derivatives provide a useful general purpose description for many kinds of signals
  - Edges
  - Corners
- In physics, objects live on a range of scales.
- Neighborhood operations can only extract local features (at a scale of at most a few pixels), however, images contain information at larger scales

Scale-Space

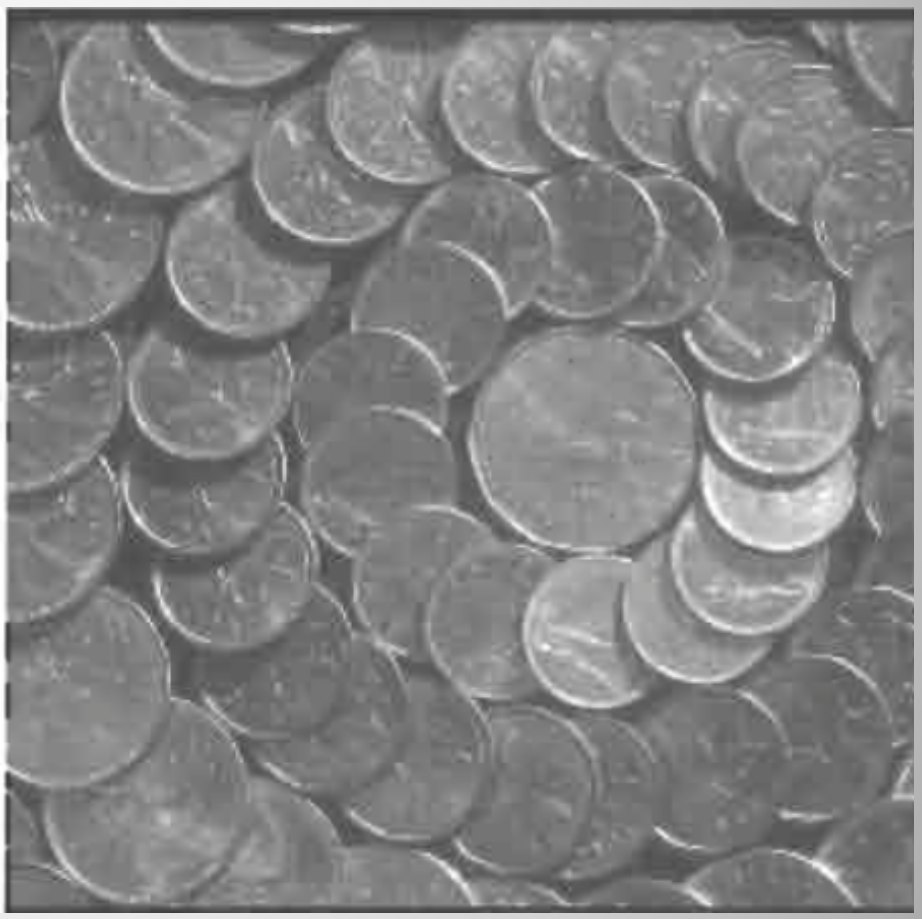
- Any given image can contain objects that exist at scales different from other objects in the same image

Scale-Space

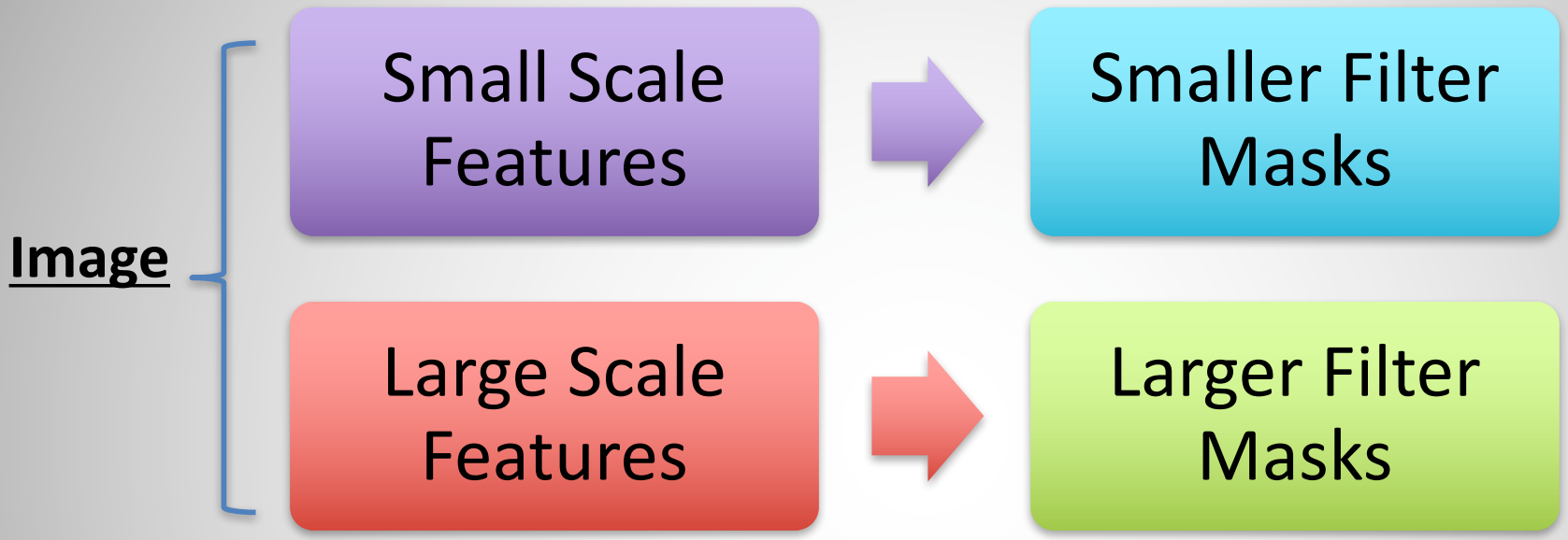
- Any given image can contain objects that exist at scales different from other objects in the same image
- Or, that even exist at multiple scales simultaneously

Scale-Space

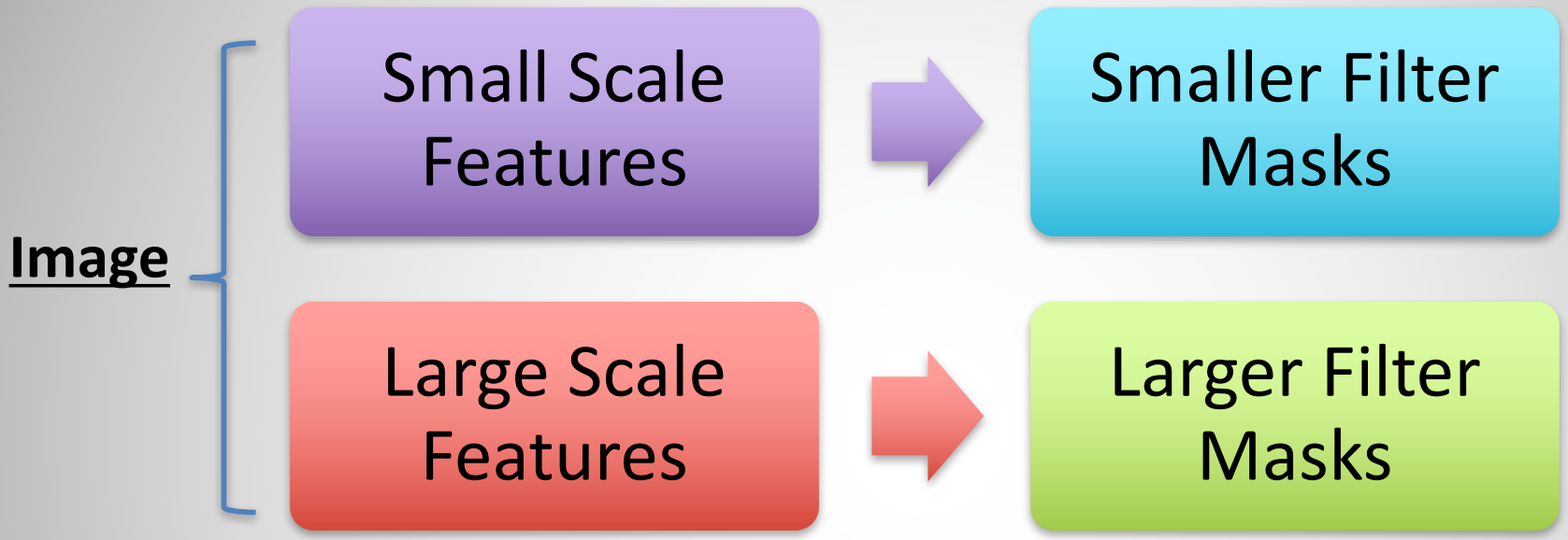
- Any given image can contain objects that exist at scales different from other objects in the same image
- Or, that even exist at multiple scales simultaneously



Multi-scale Representation



Multi-scale Representation



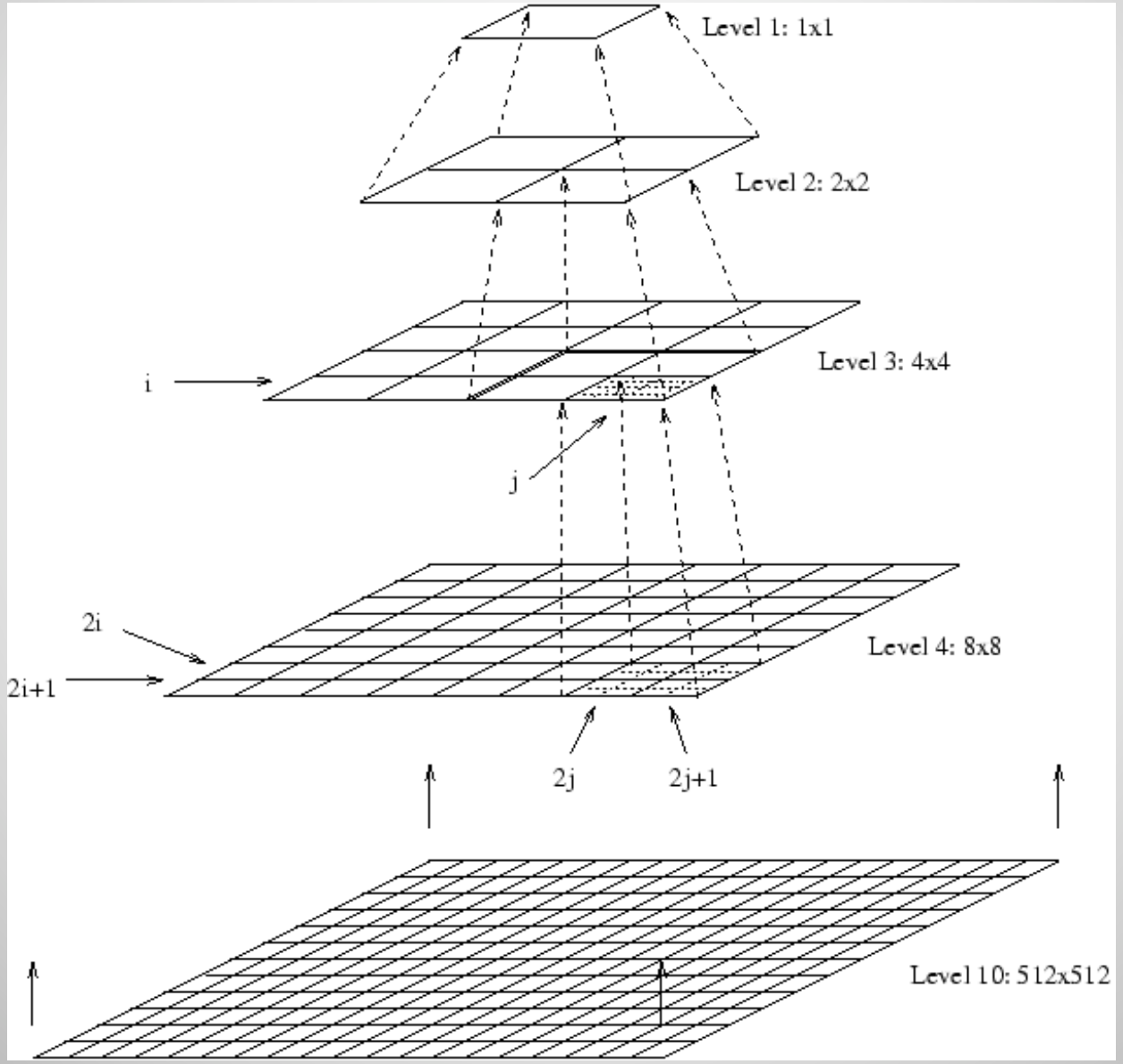
Computational Cost increases!

Doubling the scale leads to four-fold increase in the number of operations in 2D.

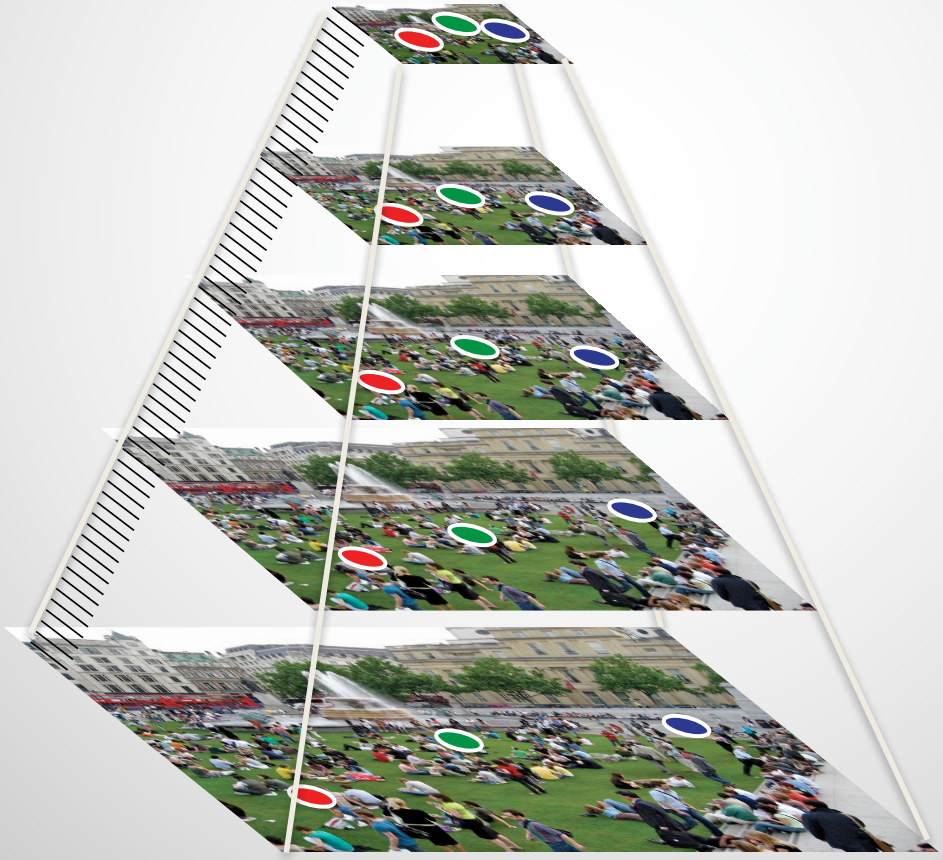
Multi-Scale Representation: Pyramid

- Pyramid is one way to represent images in multi-scale
  - Pyramid is built by using multiple copies of image.
  - Each level in the pyramid is 1/4 of the size of previous level.
  - The lowest level is of the highest resolution.
  - The highest level is of the lowest resolution.

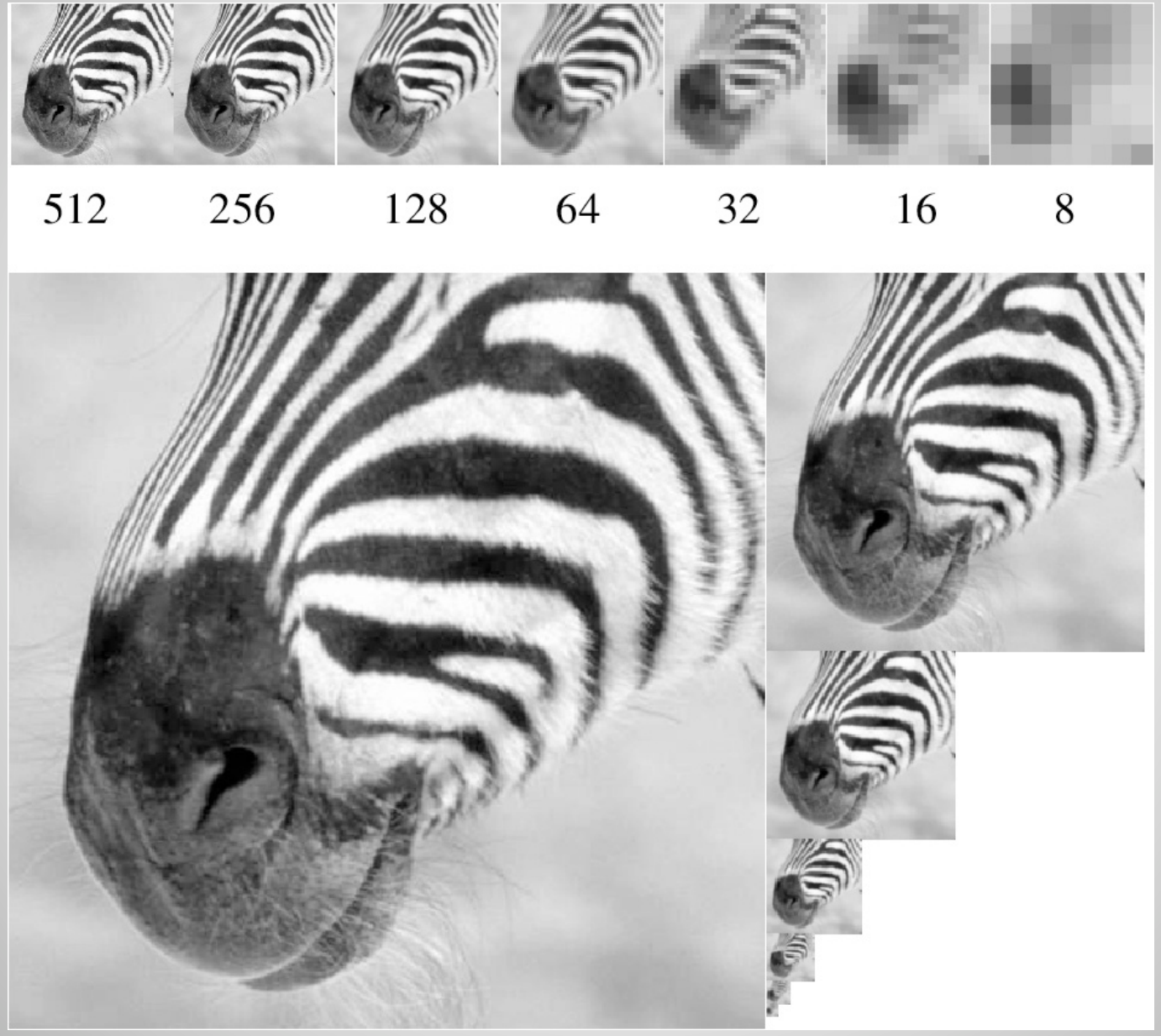
Multi-Scale Representation: Pyramid



Pyramid can capture global and local features

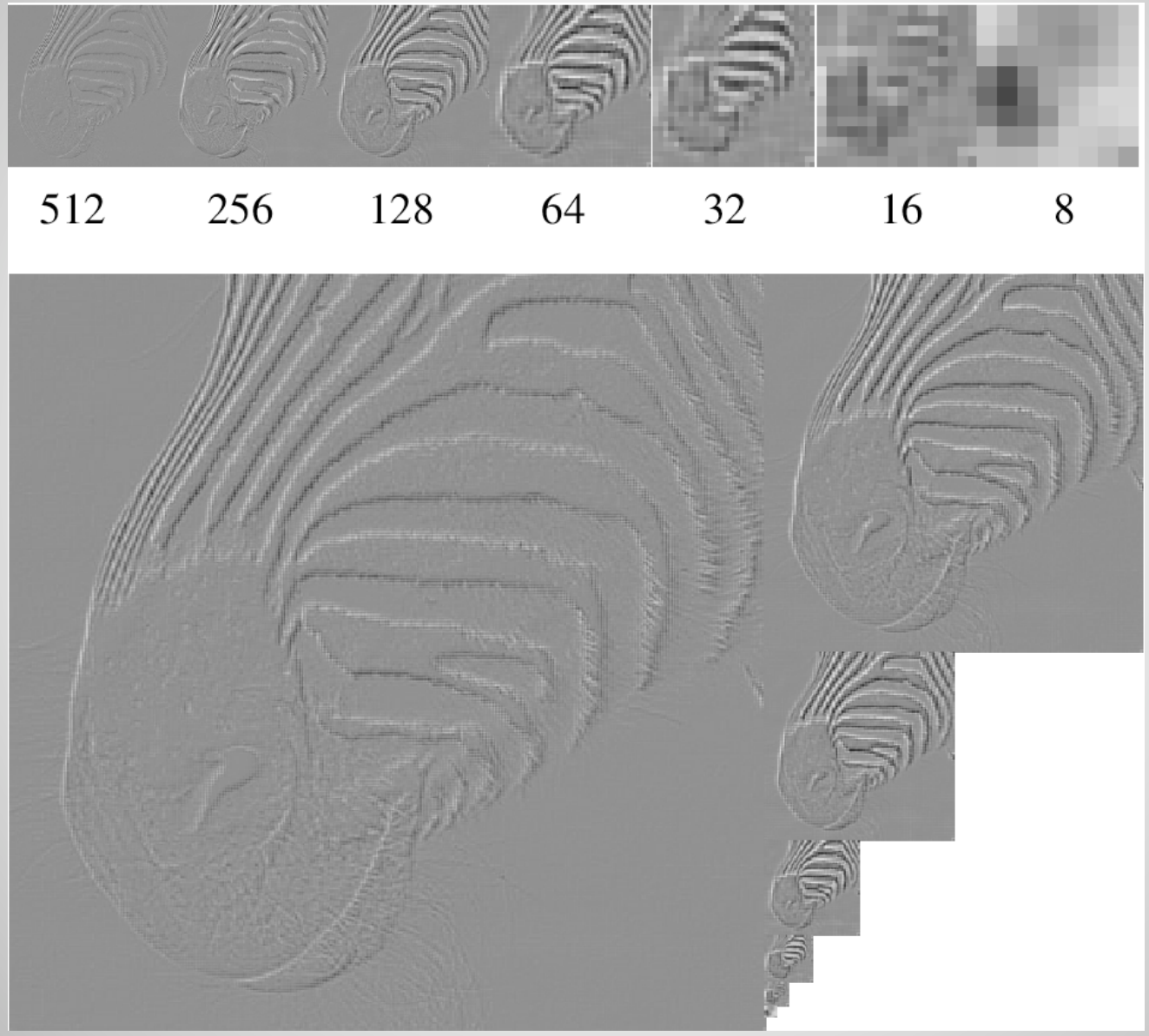


Gaussian Pyramids



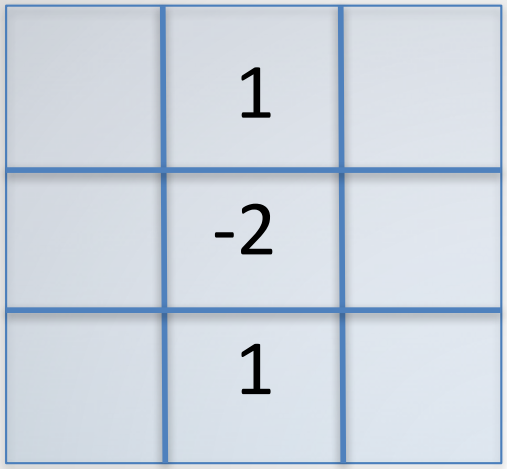
Source: Forsyth

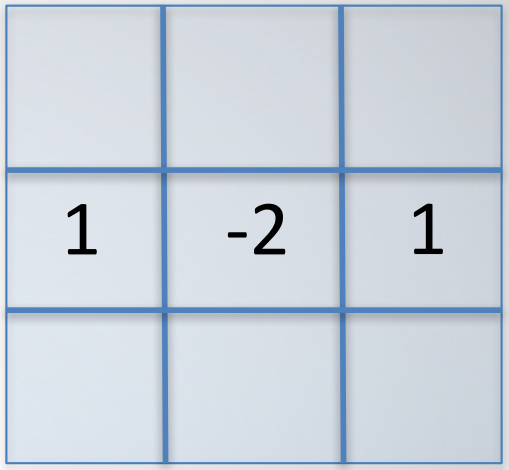
Laplacian Pyramids



Source: Forsyth

Laplacian Pyramids

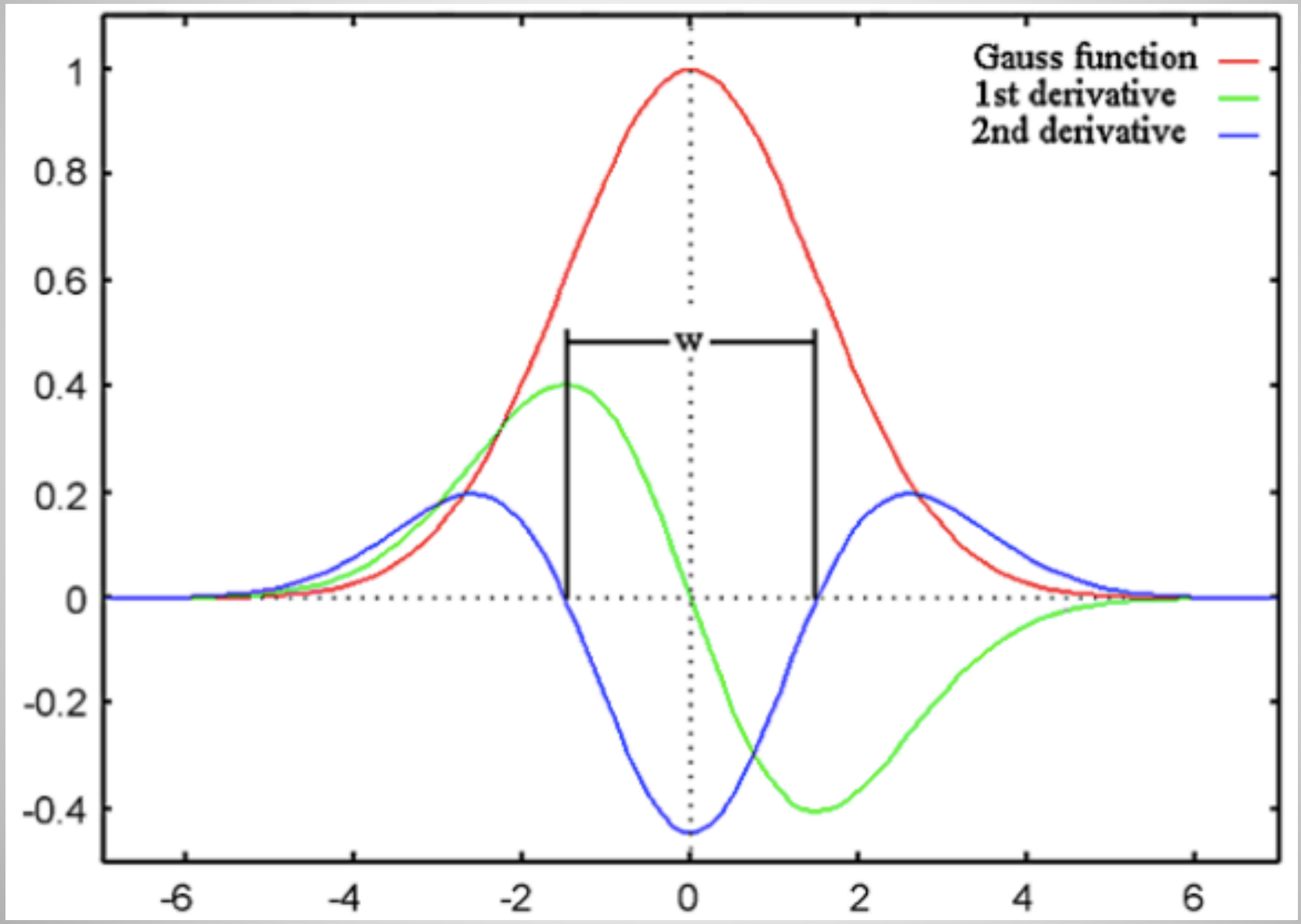




Laplacian of Gaussian (LoG): Gaussian first to smooth images then perform Laplacian operation

$$\nabla^2(G_\sigma * I) = I * \nabla^2 G_\sigma$$

Laplacian Pyramids



Laplacian Pyramids

$$\frac{\delta G_\sigma}{\delta x}(x,y) = -\frac{x}{2\pi\sigma^4} e^{-(x^2+y^2)/(2\sigma^2)}$$

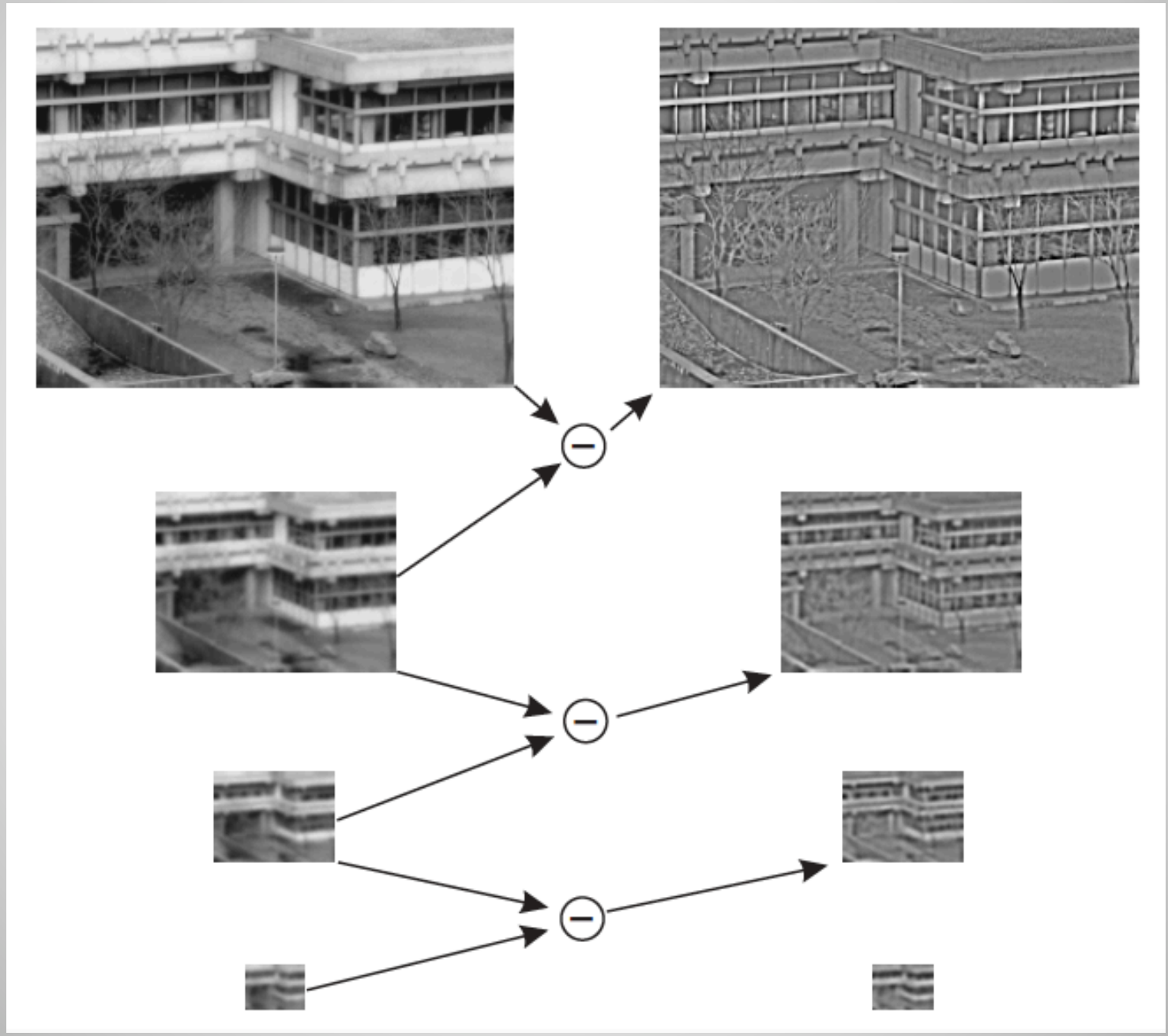
Laplacian Pyramids

$$\frac{\delta G_\sigma}{\delta x}(x,y) = -\frac{x}{2\pi\sigma^4} e^{-(x^2+y^2)/(2\sigma^2)}$$

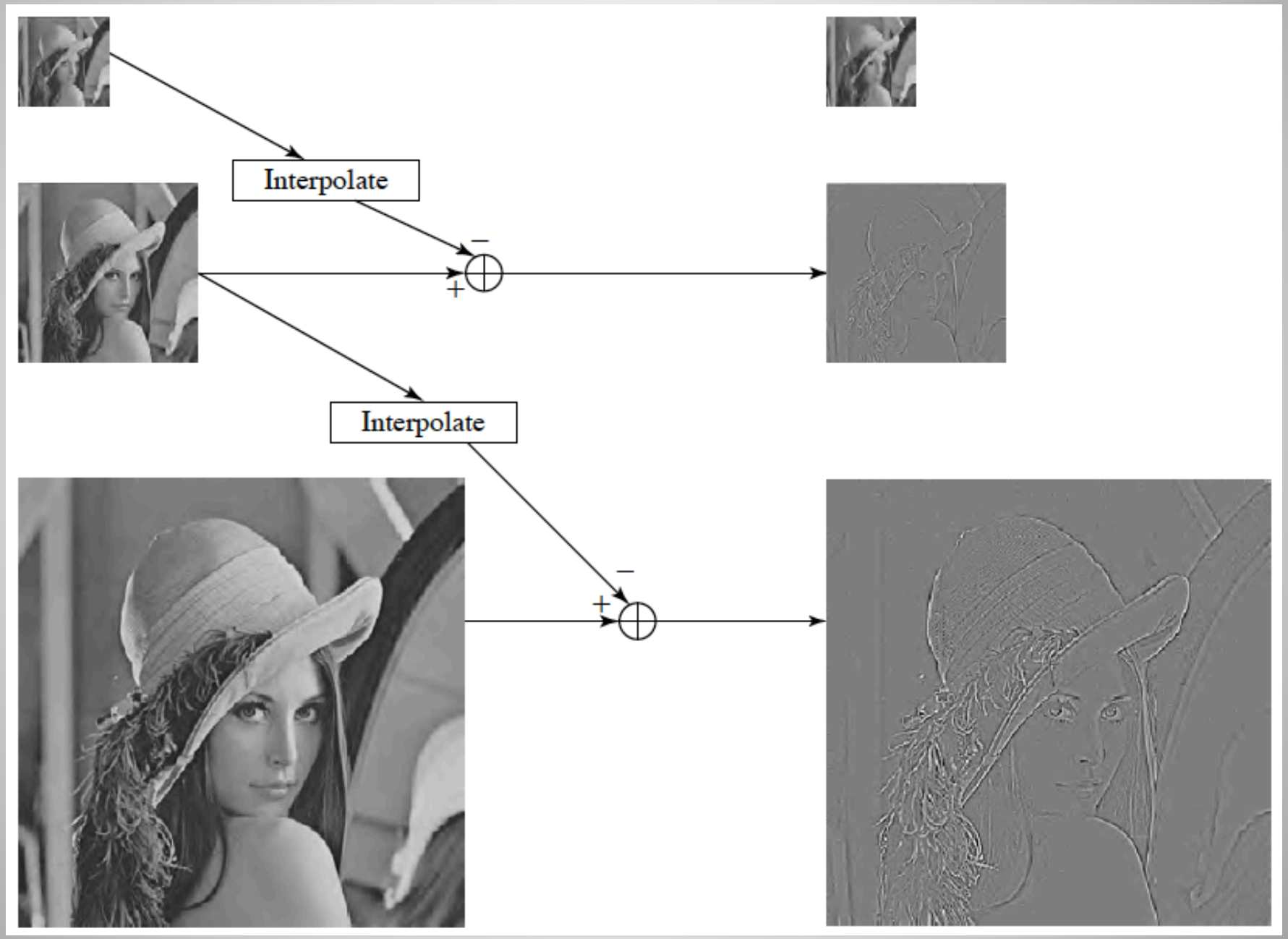
Repeat same derivative for y component, and then take second derivatives.

$$\bigtriangledown^2 G_\sigma(x,y) = \frac{1}{2\pi\sigma^4} \frac{(x^2+y^2-2\sigma^2)}{\sigma^2} e^{-(x^2+y^2)/(2\sigma^2)}$$

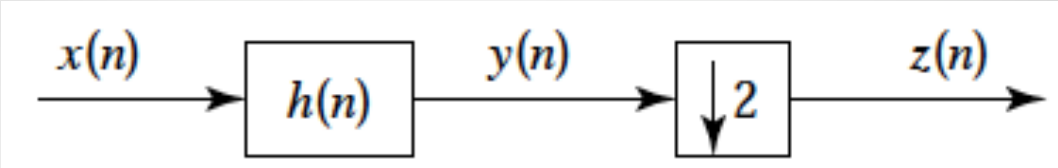
Laplacian Pyramid Construction



Laplacian Pyramid Construction

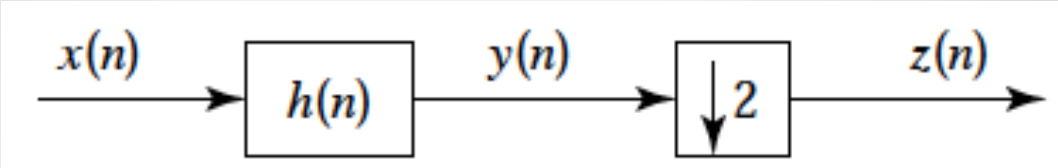


Decimation and Interpolation



LowPass filter
(i.e., Gaussian)

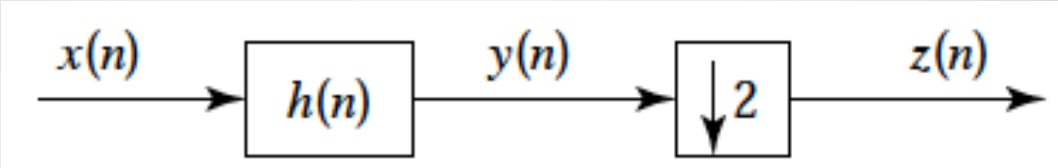
Decimation and Interpolation



LowPass filter
(i.e., Gaussian)

$$y(n) = x(n) * h(n) = \sum_k h(k)x(n-k)$$

Decimation and Interpolation

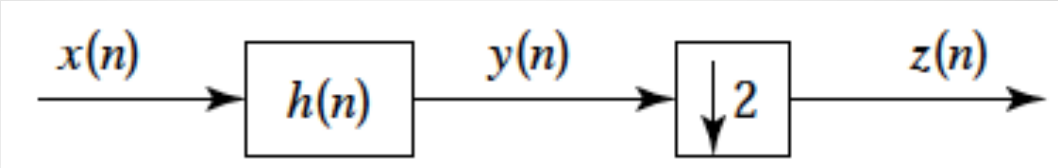


LowPass filter
(i.e., Gaussian)

$$y(n) = x(n) * h(n) = \sum_k h(k)x(n-k)$$

$$z(n) = y(2n)$$

Decimation and Interpolation



LowPass filter
(i.e., Gaussian)

$$y(n) = x(n) * h(n) = \sum_k h(k)x(n-k)$$

$$z(n) = y(2n)$$

$$z(n) = \sum_k h(k)x(2n-k)$$

Difference of Gaussian (DoG)

- It is a common approximation of LoG (better run time).

Difference of Gaussian (DoG)

- It is a common approximation of LoG (better run time).

$$D_{\sigma,\alpha}(x,y) = L(x,y,\alpha) - L(x,y,\alpha\sigma)$$

Difference of Gaussian (DoG)

- It is a common approximation of LoG (better run time).

$$D_{\sigma,\alpha}(x,y) = L(x,y,\alpha) - L(x,y,\alpha\sigma)$$

- It is the difference between a blurred copy of image I and an even more blurred copy of I.

Difference of Gaussian (DoG)

- It is a common approximation of LoG (better run time).

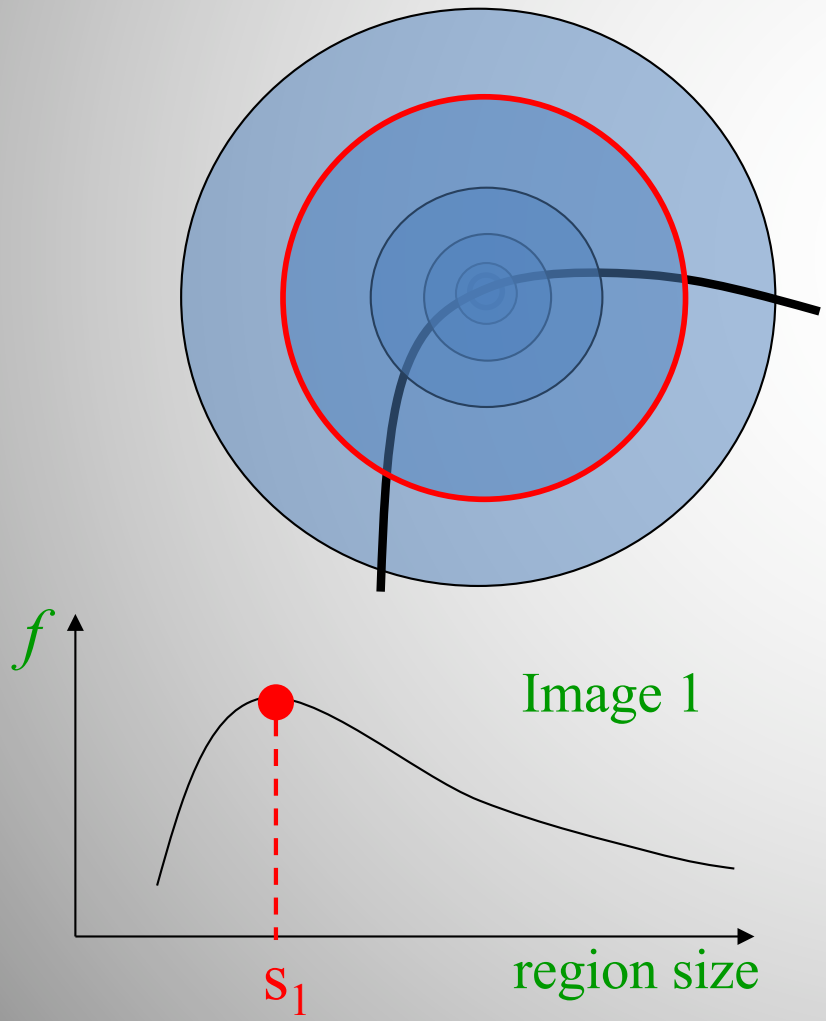
$$D_{\sigma,\alpha}(x,y) = L(x,y,\alpha) - L(x,y,\alpha\sigma)$$

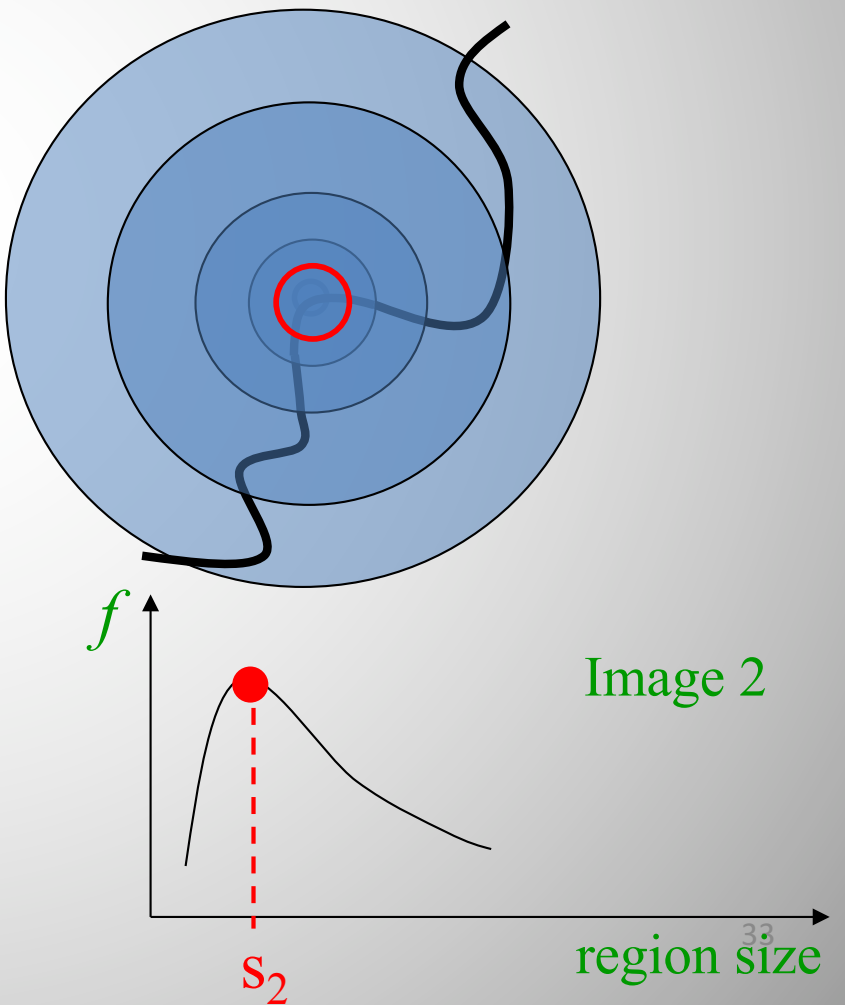
- It is the difference between a blurred copy of image I and an even more blurred copy of I.

$$\bigtriangledown G_\sigma(x,y) \approx \frac{G_{\alpha\sigma}(x,y) - G_\sigma(x,y)}{(\alpha-1)\sigma^2}$$

Scale Selection-Automated

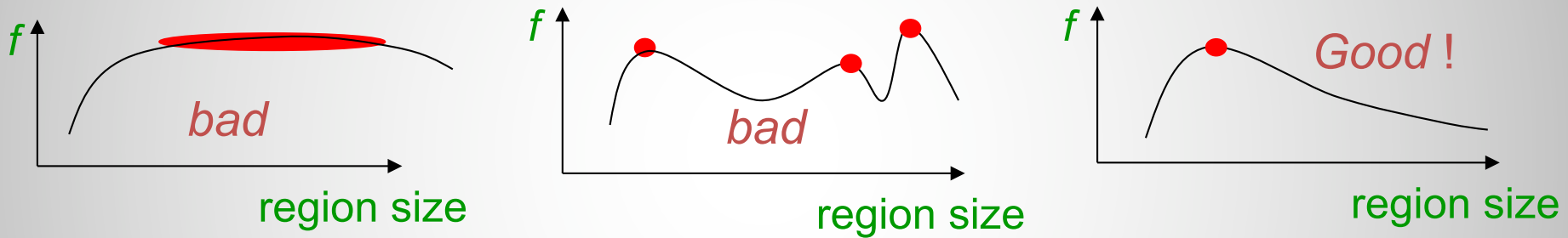
- Find scale that gives local maxima of some function *f* in both position and scale.





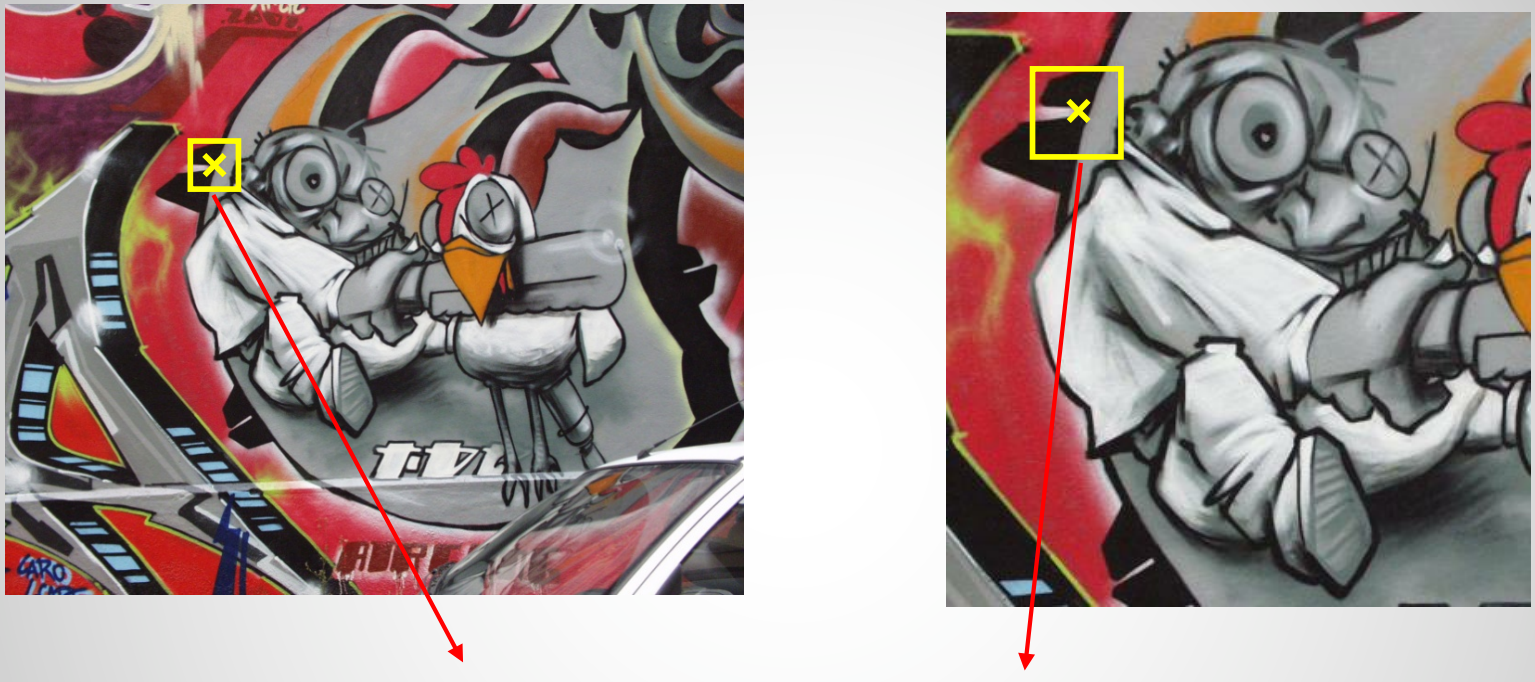
Good Function?

- A "good" function for scale detection: has one stable sharp peak



- For usual images: a good function would be a one which responds to contrast **(sharp local intensity change)**

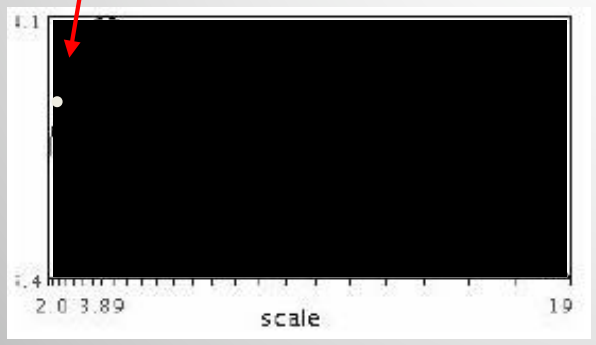
Patch Size Corresponding to Scale



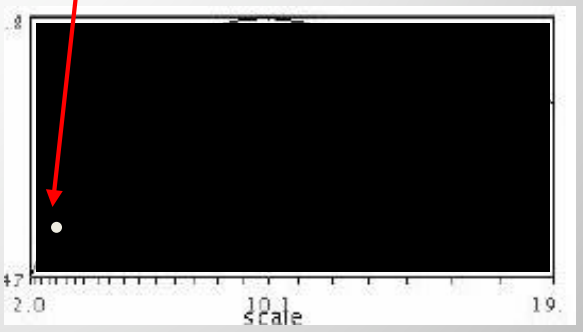
$$f(I_{i_1 \ldots i_m}(x,\sigma)) \quad = \quad f(I_{i_1 \ldots i_m}(x',\sigma'))$$

How to find corresponding patch sizes?

Patch Size Corresponding to Scale

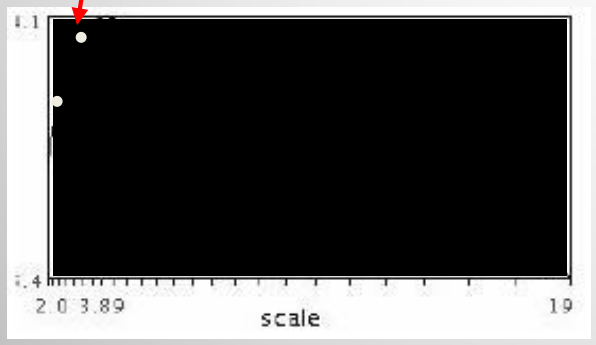


$f(I_{i_1 \ldots i_m}(x,\sigma))$

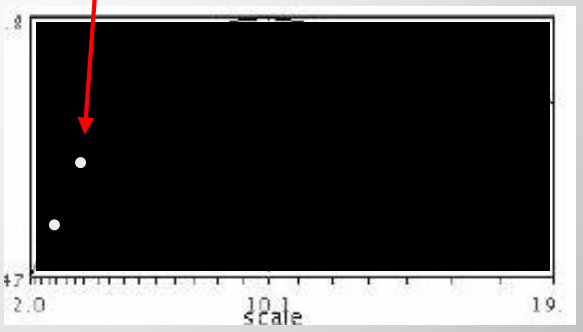


$f(I_{i_1 \ldots i_m}(x',\sigma))$

Patch Size Corresponding to Scale

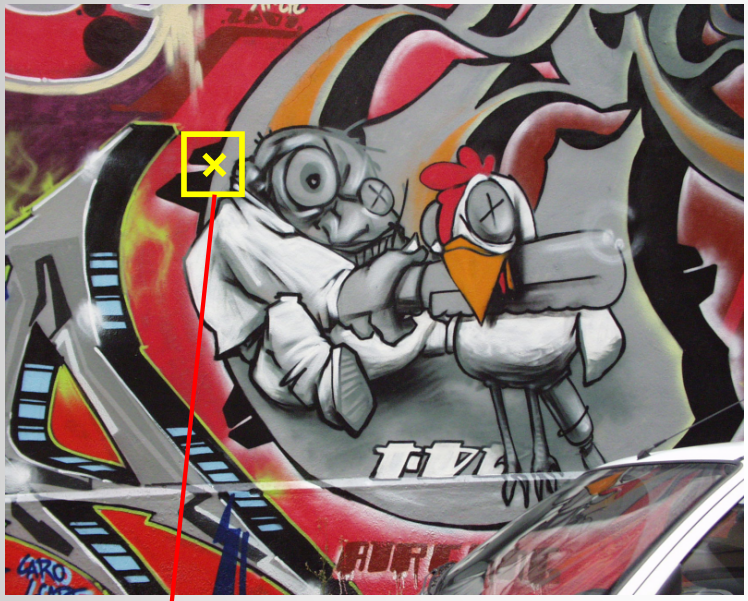


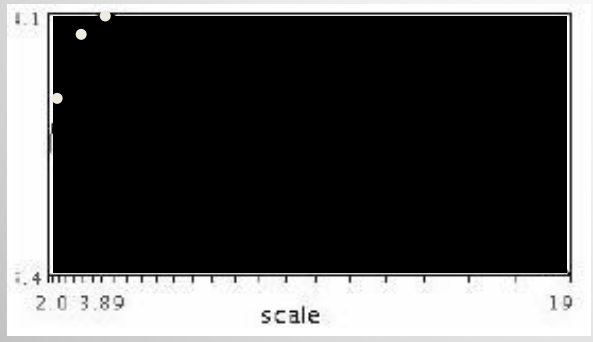
$f(I_{i_1 \ldots i_m}(x,\sigma))$



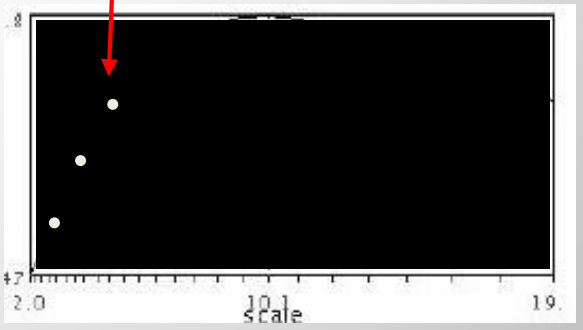
$f(I_{i_1 \ldots i_m}(x',\sigma))$

Patch Size Corresponding to Scale



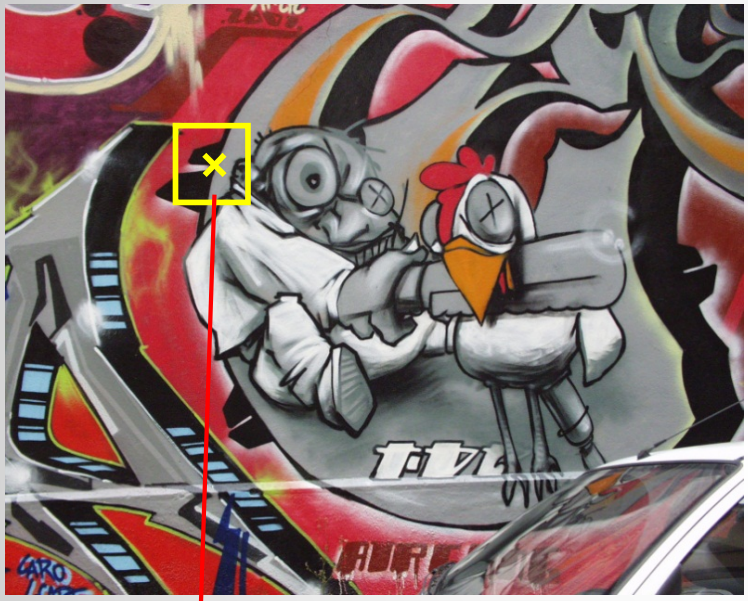


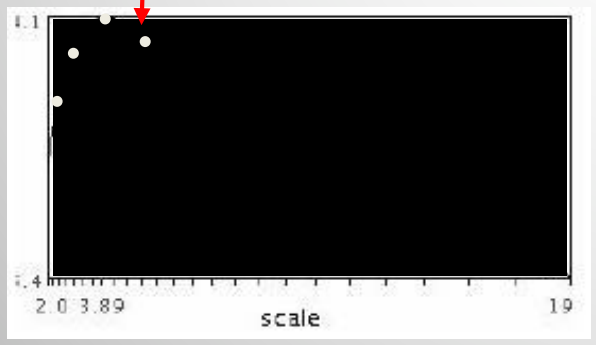
$f(I_{i_1 \ldots i_m}(x,\sigma))$



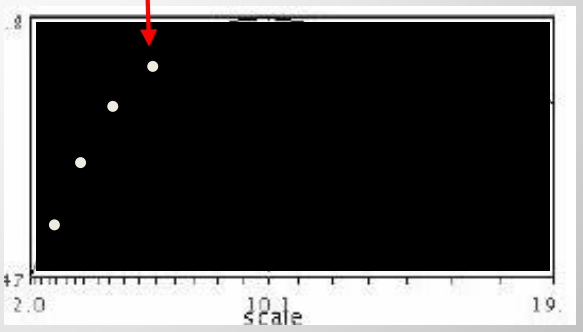
$f(I_{i_1 \ldots i_m}(x',\sigma))$

Patch Size Corresponding to Scale



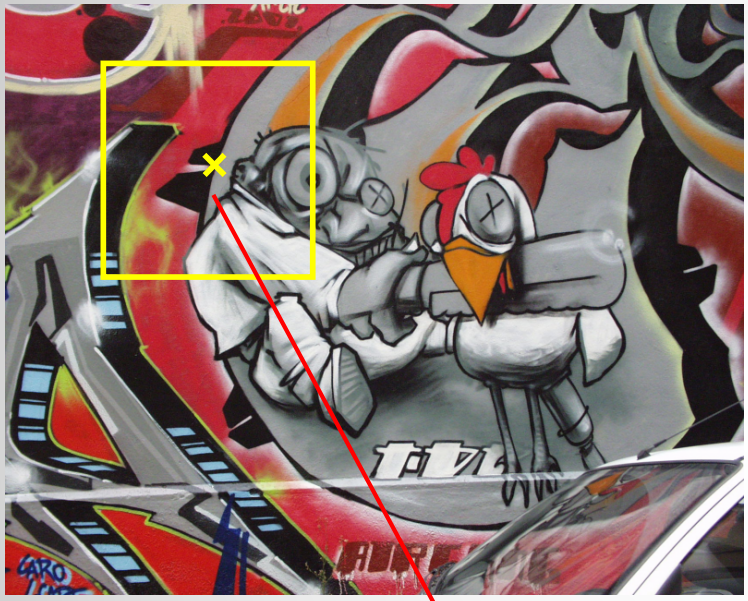


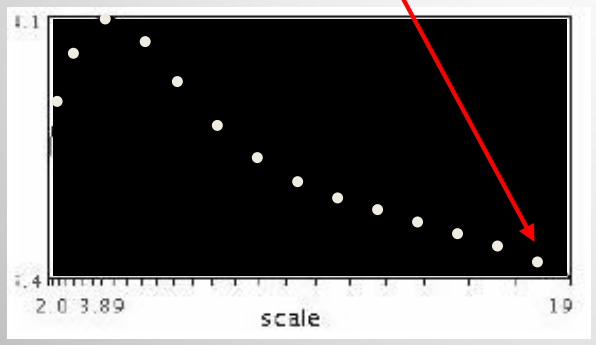
$f(I_{i_1 \ldots i_m}(x,\sigma))$



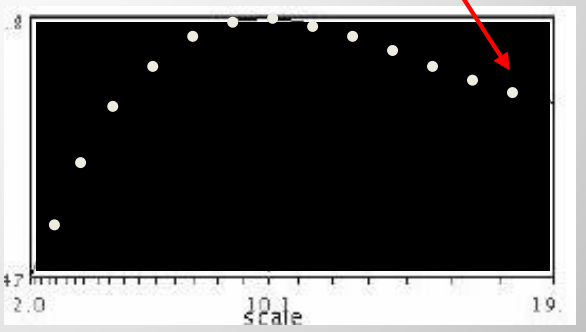
$f(I_{i_1 \ldots i_m}(x',\sigma))$

Patch Size Corresponding to Scale





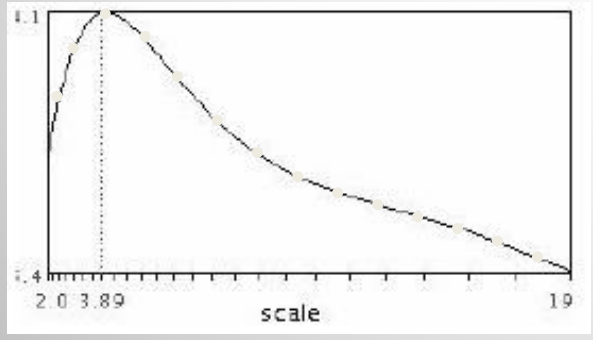
$f(I_{i_1 \ldots i_m}(x,\sigma))$



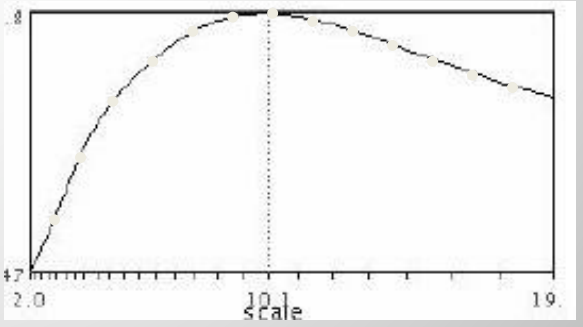
$f(I_{i_1 \ldots i_m}(x',\sigma))$

Patch Size Corresponding to Scale

- Function responses for increasing scale (scale signature)

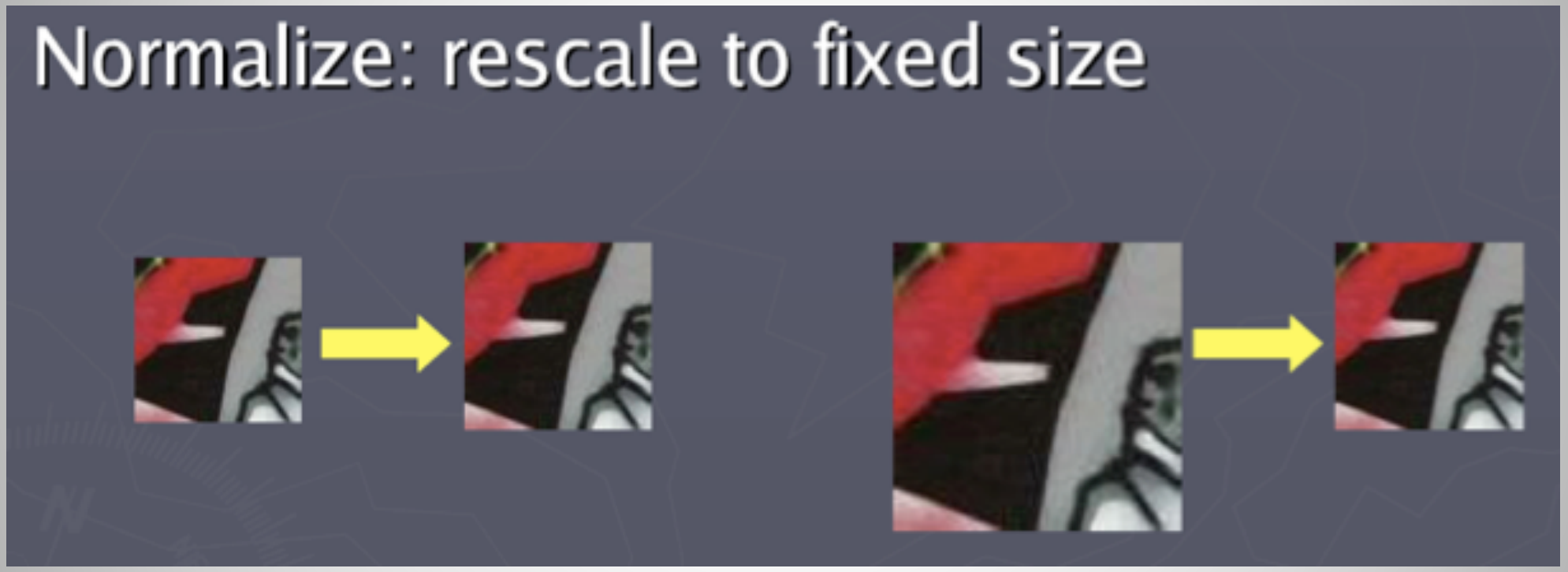


$f(I_{i_1 \ldots i_m}(x,\sigma))$



$f(I_{i_1 \ldots i_m}(x',\sigma'))$

Normalization



Scale Invariant Feature Transform (SIFT)

- Lowe., D. 2004, IJCV

cited > 43K

Distinctive Image Features from Scale-Invariant Keypoints

DAVID G. LOWE
Computer Science Department, University of British Columbia, Vancouver, B.C., Canada
Lowe@cs.ubc.ca

Received January 10, 2003; Revised January 7, 2004; Accepted January 22, 2004

Abstract. This paper presents a method for extracting distinctive invariant features from images that can be used to perform reliable matching between different views of an object or scene. The features are invariant to image scale and rotation, and are shown to provide robust matching across a substantial range of affine distortion, change in 3D viewpoint, addition of noise, and change in illumination. The features are highly distinctive, in the sense that a single feature can be correctly matched with high probability against a large database of features from many images. This paper also describes an approach to using these features for object recognition. The recognition proceeds by matching individual features to a database of features from known objects using a fast nearest-neighbor algorithm, followed by a Hough transform to identify clusters belonging to a single object, and finally performing verification through least-squares solution for consistent pose parameters. This approach to recognition can robustly identify objects among clutter and occlusion while achieving near real-time performance.

Keywords: invariant features, object recognition, scale invariance, image matching

1. Introduction

Image matching is a fundamental aspect of many problems in computer vision, including object or scene recognition, solving for 3D structure from multiple images, stereo correspondence, and motion tracking. This paper describes image features that have many properties that make them suitable for matching differing images of an object or scene. The features are invariant to image scaling and rotation, and partially invariant to change in illumination and 3D camera viewpoint. They are well localized in both the spatial and frequency domains, reducing the probability of disruption by occlusion, clutter, or noise. Large numbers of features can be extracted from typical images with efficient algorithms. In addition, the features are highly distinctive, which allows a single feature to be correctly matched with high probability against a large database of features, providing a basis for object and scene recognition.

The cost of extracting these features is minimized by taking a cascade filtering approach, in which the more expensive operations are applied only at locations that pass an initial test. Following are the major stages of computation used to generate the set of image features:

1. *Scale-space extrema detection*: The first stage of computation searches over all scales and image locations. It is implemented efficiently by using a difference-of-Gaussian function to identify potential interest points that are invariant to scale and orientation.
2. *Keypoint localization*: At each candidate location, a detailed model is fit to determine location and scale. Keypoints are selected based on measures of their stability.
3. *Orientation assignment*: One or more orientations are assigned to each keypoint location based on local image gradient directions. All future operations are performed on image data that has been transformed relative to the assigned orientation, scale, and location for each feature, thereby providing invariance to these transformations.

Scale Invariant Feature Transform (SIFT)

- Shows existence, importance, and value of invariant types of detector
- Demonstrates the richness that feature descriptors can bring to feature matching

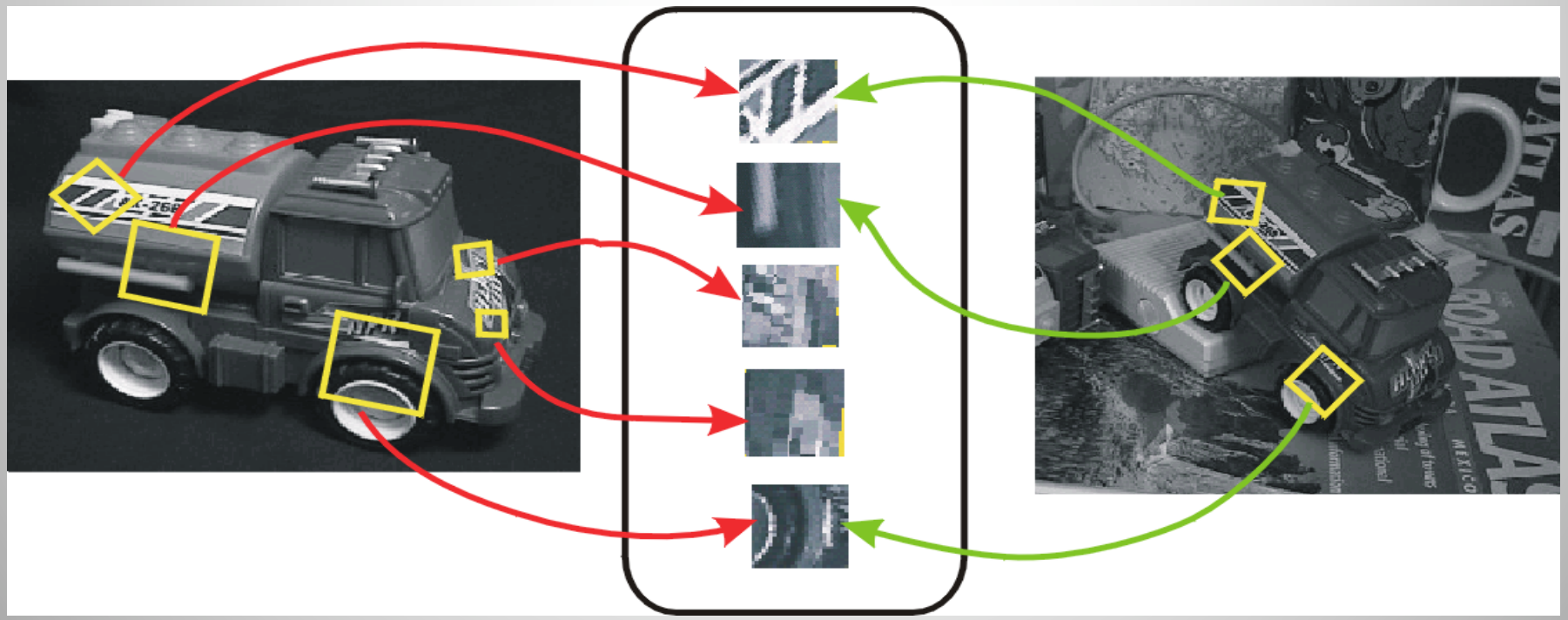
Instead of using LoG (Laplacian of Gaussian) like in Harris and Hessian-based operators, SIFT uses DoG (Difference of Gaussian).

Scale Invariant Feature Transform (SIFT)

- Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters

Scale Invariant Feature Transform (SIFT)

- Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters



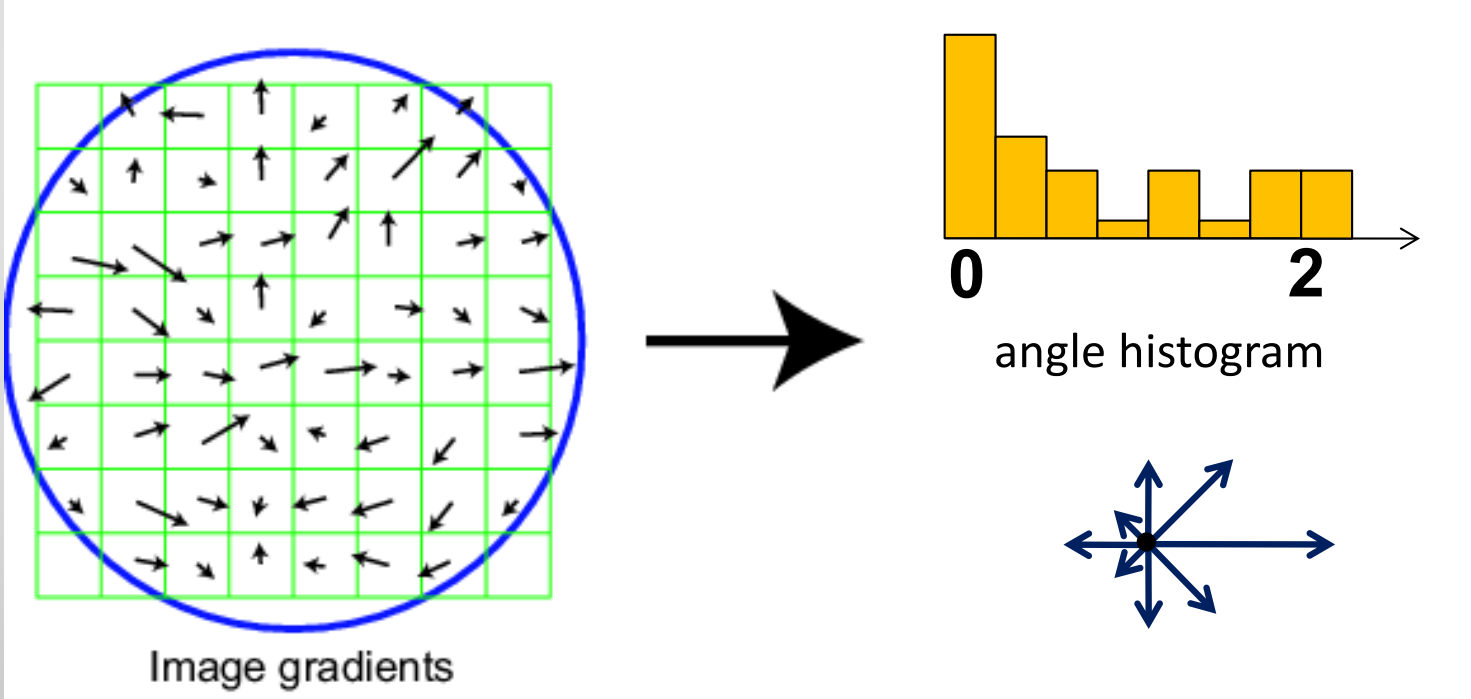
Overall Procedure at a High Level

Scale-Space Extrema Detection	Search over multiple scales and image locations
KeyPoint Localization	Fit a model to determine location and scale. Select KeyPoints based on a measure of stability.
Orientation Assignment	Compute best orientation(s) for each keyPoint region.
KeyPoint Description	Use local image gradients at selected scale and rotation to describe each keyPoint region.

SIFT Descriptor

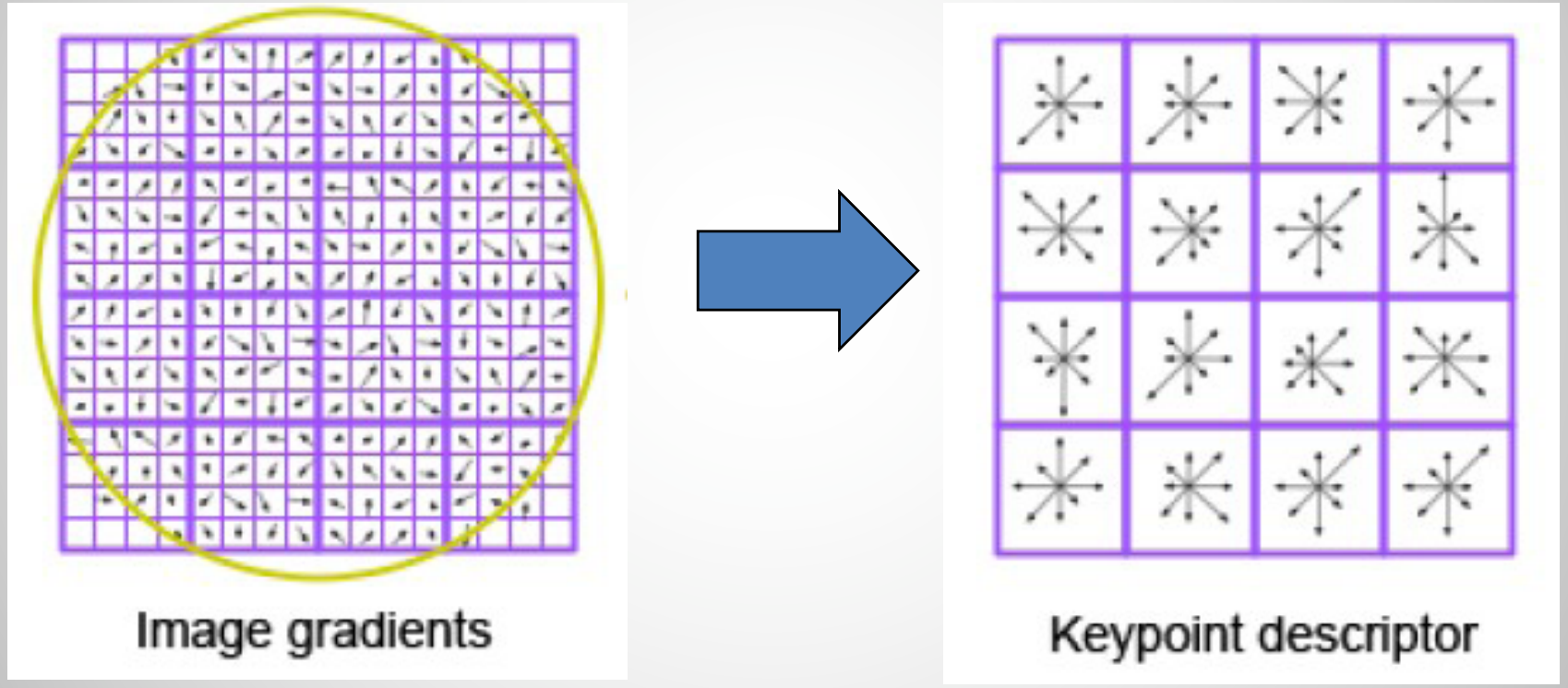
Basic idea:

- Take n x n (i.e., n=16) square window (around a *feature/interest point*)
- Divide them into m x m (i.e., m=4) cells
- Compute gradient orientation for each cell
- Create histogram over edge orientations weighted by magnitude



SIFT Descriptor

16 histograms x 8 orientations
= 128 features

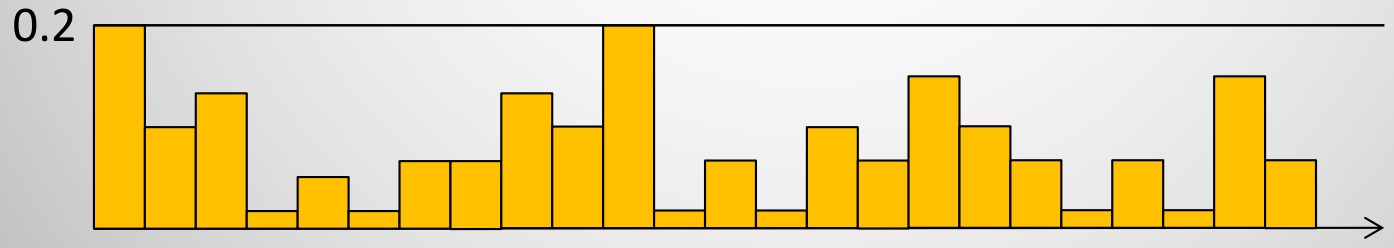


SIFT Descriptor

Full version

- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
- Compute an orientation histogram for each cell
- 16 cells * 8 orientations = 128 dimensional descriptor
- Threshold normalize the descriptor:

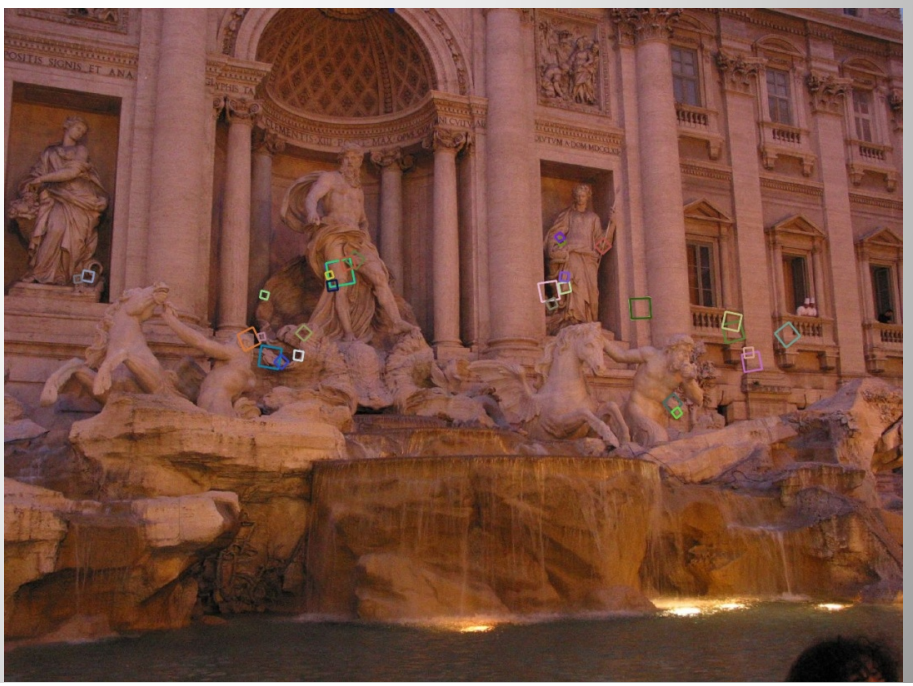
$$\sum_i d_i^2 = 1 \quad \text{such that:} \quad d_i < 0.2$$



Properties of SIFT

Extraordinarily robust matching technique

- Can handle changes in viewpoint
- Can handle significant changes in illumination
- Efficient (real time)
- Lots of code available
  - http://people.csail.mit.edu/albert/ladypack/wiki/index.php/Known_implementations_of_SIFT



Revisit SIFT 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.

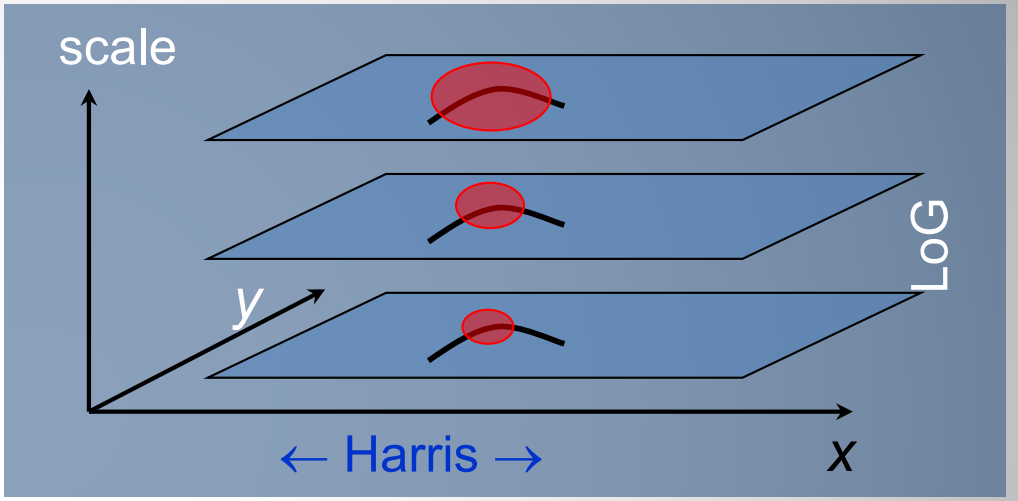
(1) Scale-Space Extrema Detection

- We want to find points that give us information about the objects in the image.
- The information about the objects is in the object's **edges**.
- We will represent the image in a way that gives us these edges as this representations extrema points.

(1) Scale-Space Extrema Detection

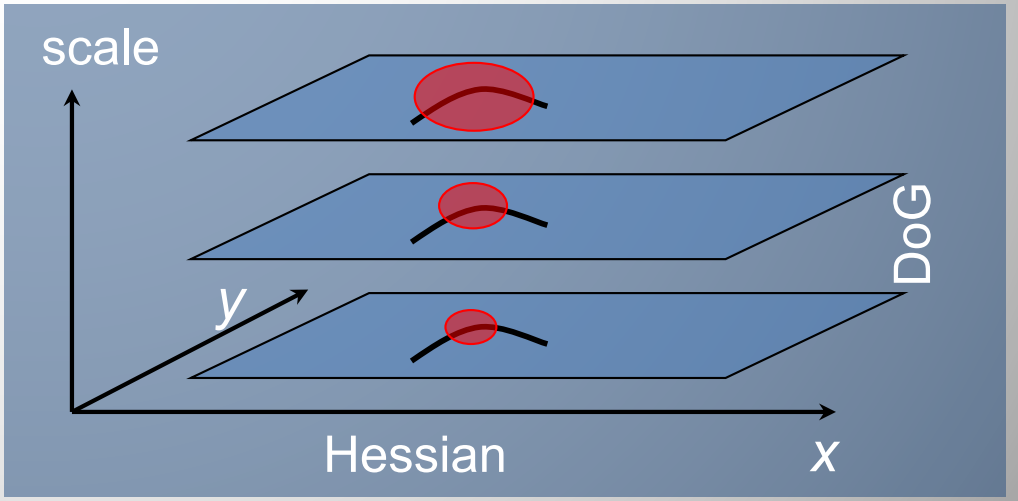
- Harris-Laplace

Find local maxima of:
- Harris detector in space
- LoG in scale

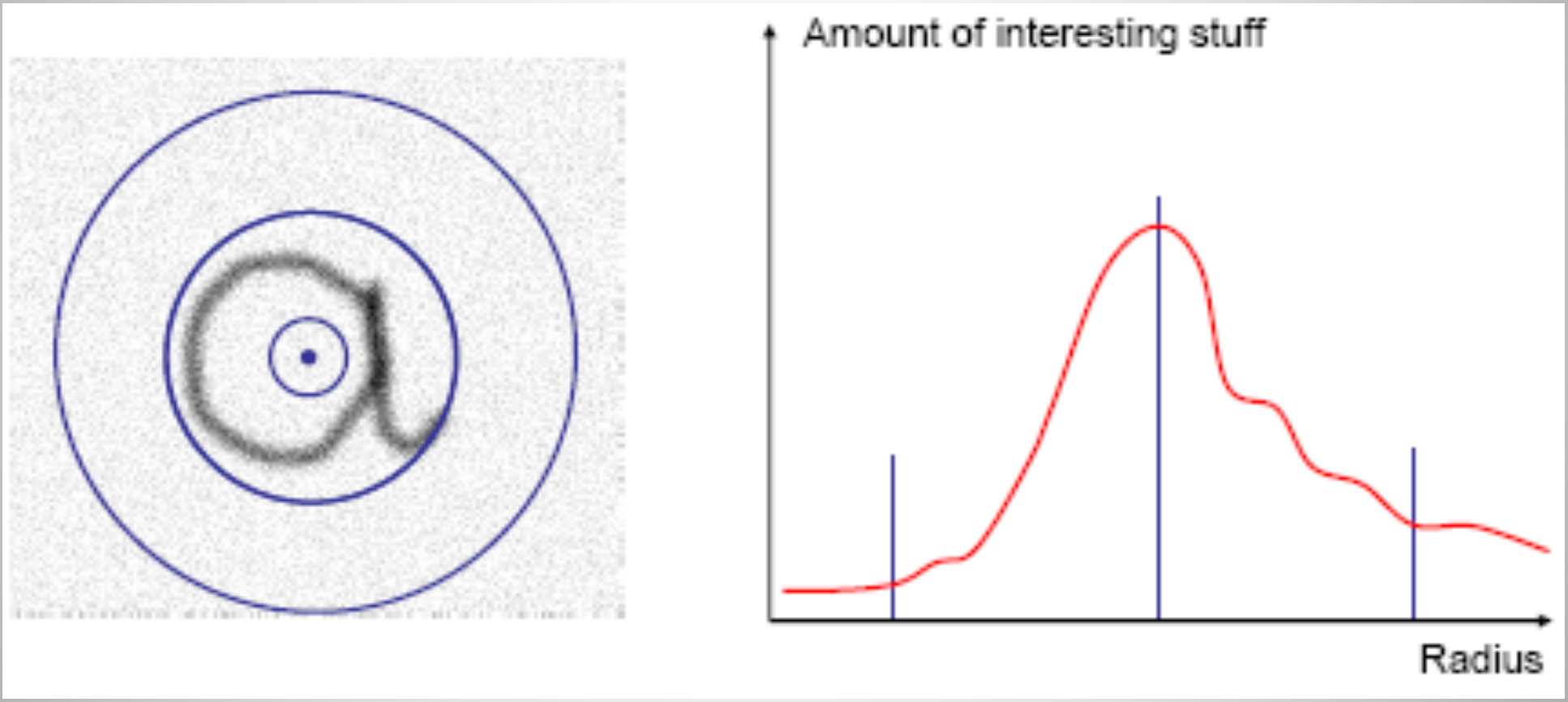


- SIFT

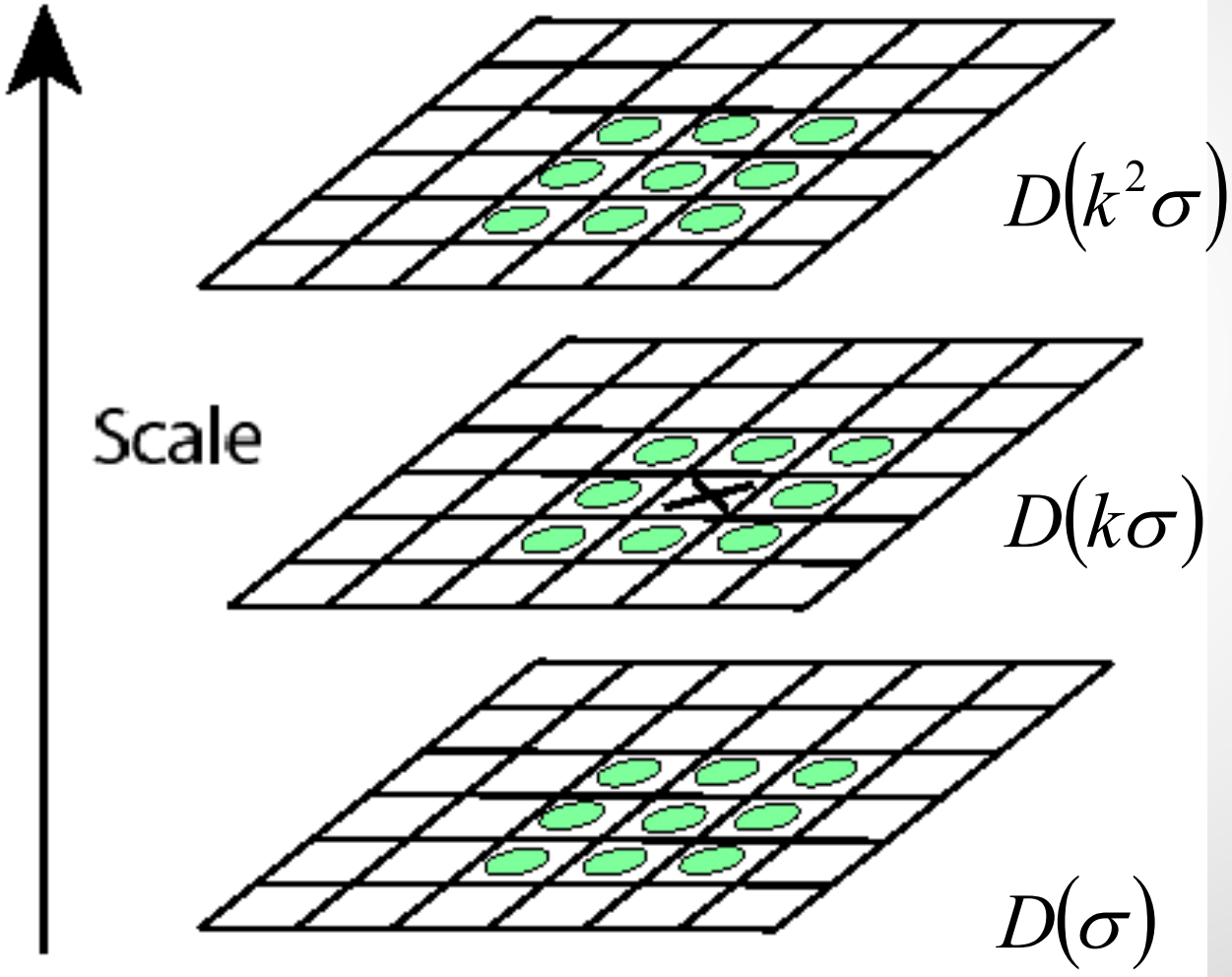
Find local maxima of:
- Hessian in space
- DoG in scale



(1) Scale-Space Extrema Detection



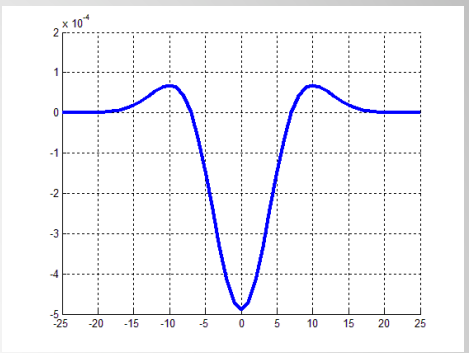
(1) Scale-Space Extrema Detection

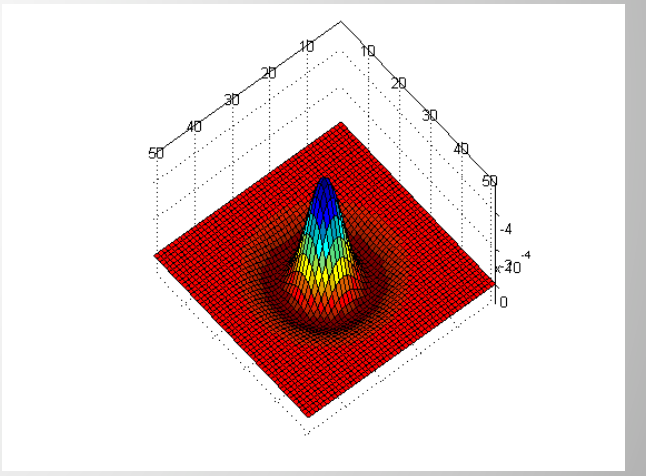


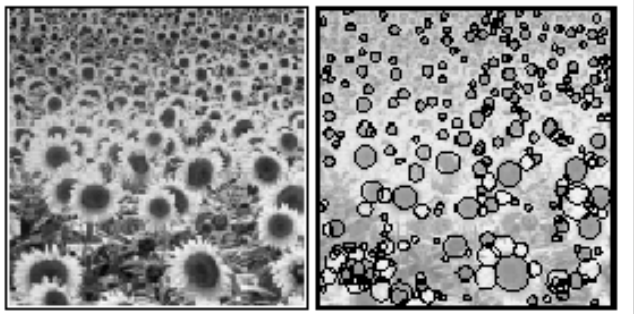
- Extract local extrema (i.e., minima or maxima) in DoG pyramid.
  - -Compare each point to its 8 neighbors at the same level, 9 neighbors in the level above, and 9 neighbors in the level below (i.e., 26 total).

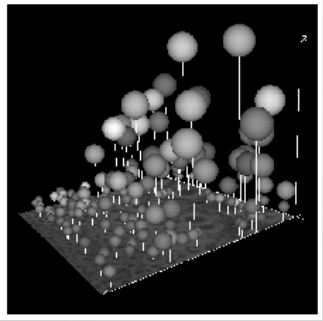
(1) Scale-Space Extrema Detection

- **Laplacian of Gaussian** kernel
  - Scale normalized (x by scale^2)
  - Proposed by Lindeberg
- Scale-space detection
  - Find local maxima across scale/space
  - A good “blob” detector









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

$$\nabla^2 G(x, y, \sigma) = \frac{\partial^2 G}{\partial x^2} + \frac{\partial^2 G}{\partial y^2}$$

(2) KeyPoint Localization

- There are still a lot of points, some of them are not good enough.
- The locations of keypoints may be not accurate.
- Eliminating edge points.

(2) KeyPoint Localization

- Reject (1) points with low contrast (flat)
  (2) poorly localized along an edge (edge)

(2) KeyPoint Localization

- Reject (1) points with low contrast (flat)
  (2) poorly localized along an edge (edge)

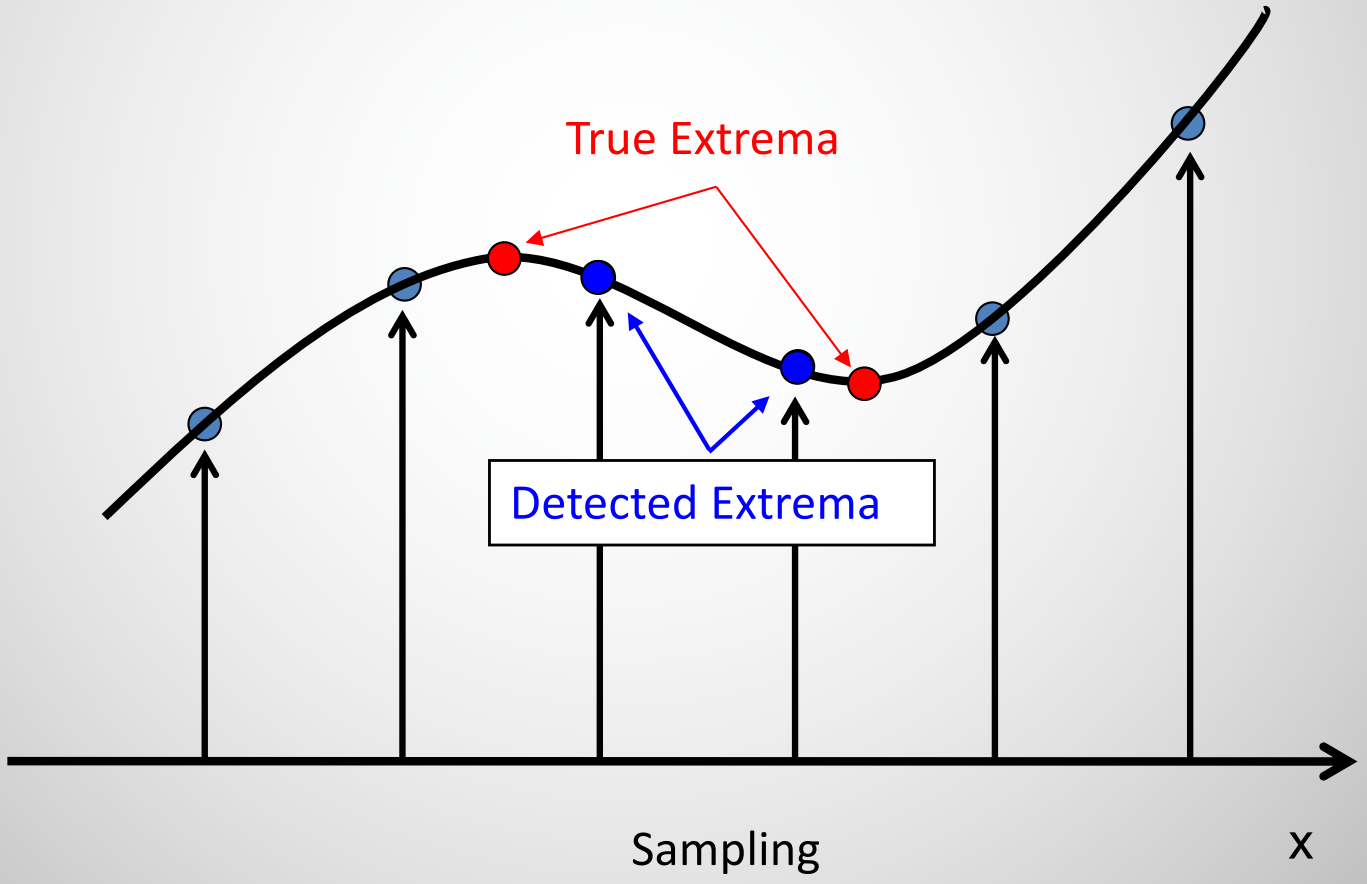
if $$\frac{Tr(H)}{|H|} < \frac{(r+1)^2}{r}$$

Reject

Where $$r = \frac{\lambda_1}{\lambda_2}$$ and practically SIFT uses r=10.

Inaccurate KeyPoint Localization

- Poor contrast



Inaccurate KeyPoint Localization

- The Solution:
  - Taylor expansion:

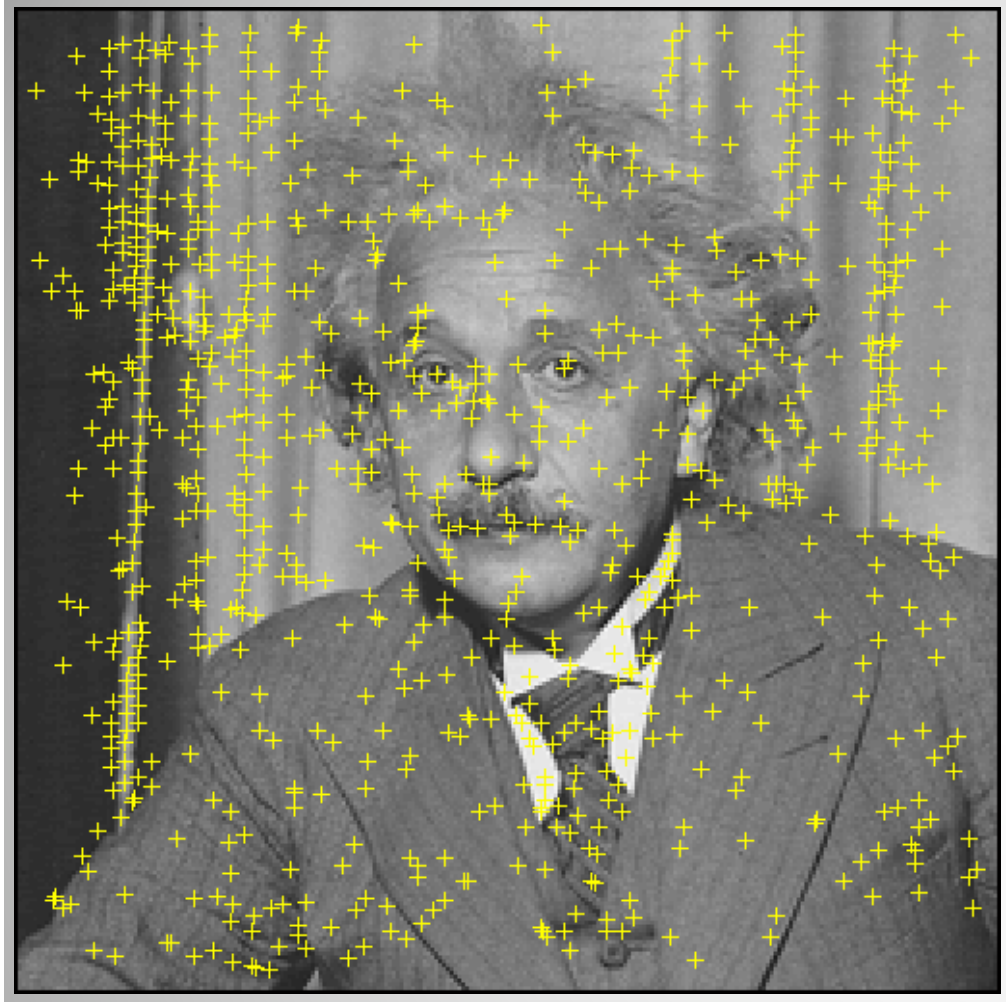
$$D(\vec{x}) = D + \frac{\partial D^T}{\partial \vec{x}} \vec{x} + \frac{1}{2} \vec{x}^T \frac{\partial^2 D^T}{\partial \vec{x}^2} \vec{x}$$

  - Minimize to find accurate extrema:

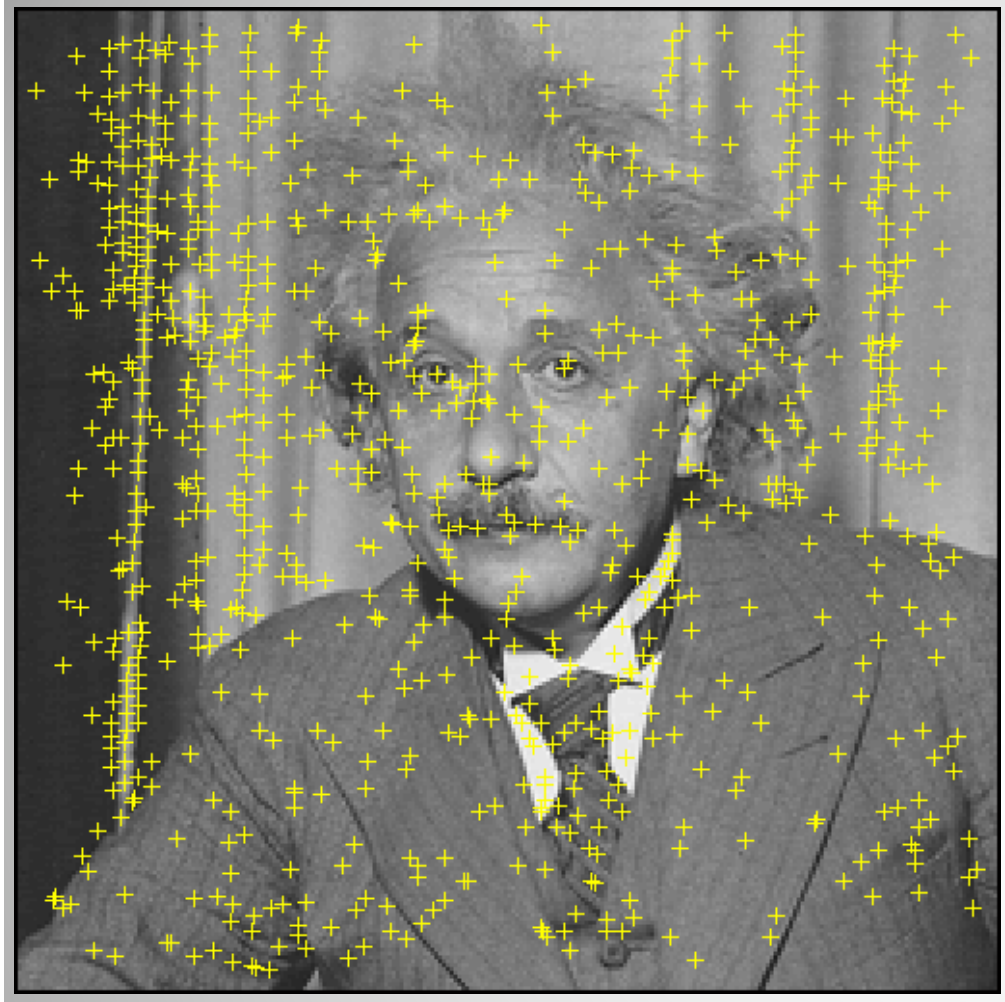
$$\hat{x} = -\frac{\partial^2 D}{\partial \vec{x}^2}^{-1} \frac{\partial D}{\partial \vec{x}}$$

  - If offset from sampling point is larger than 0.5 - Keypoint should be in a different sampling point.

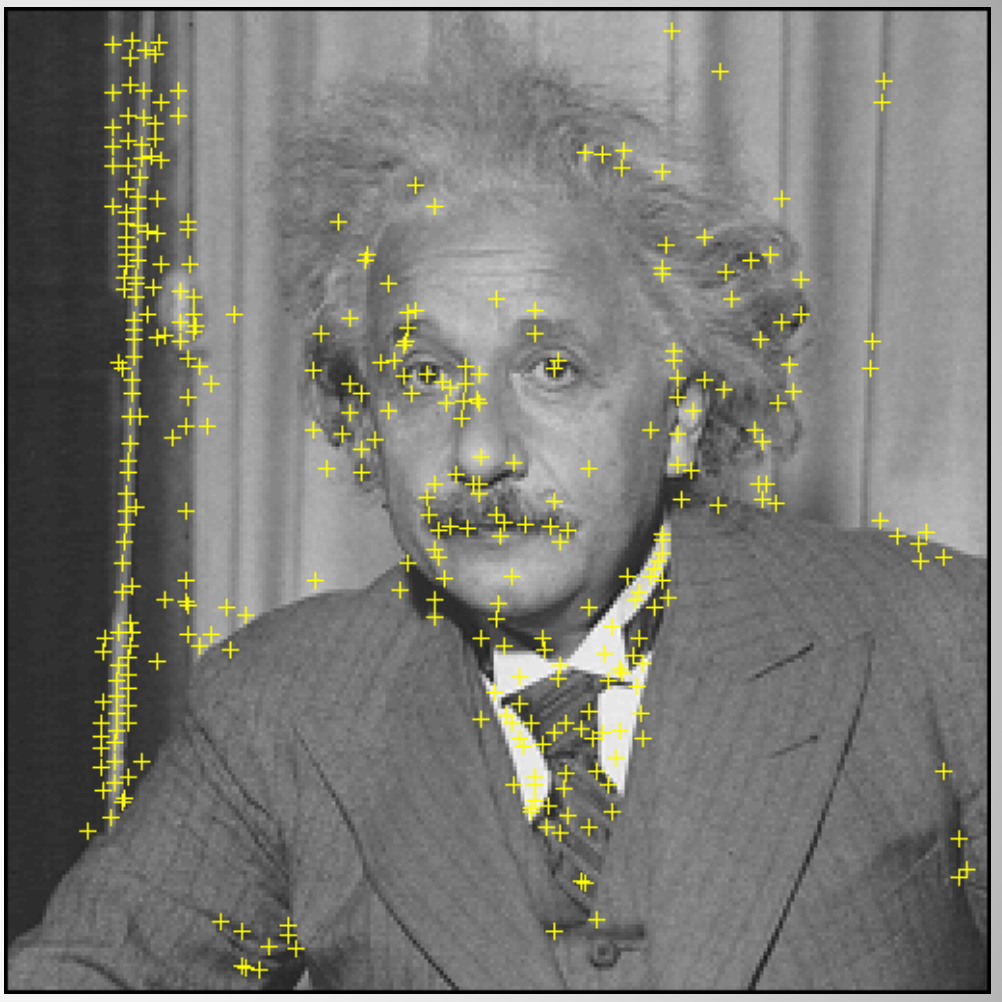
Brown & Lowe 2002



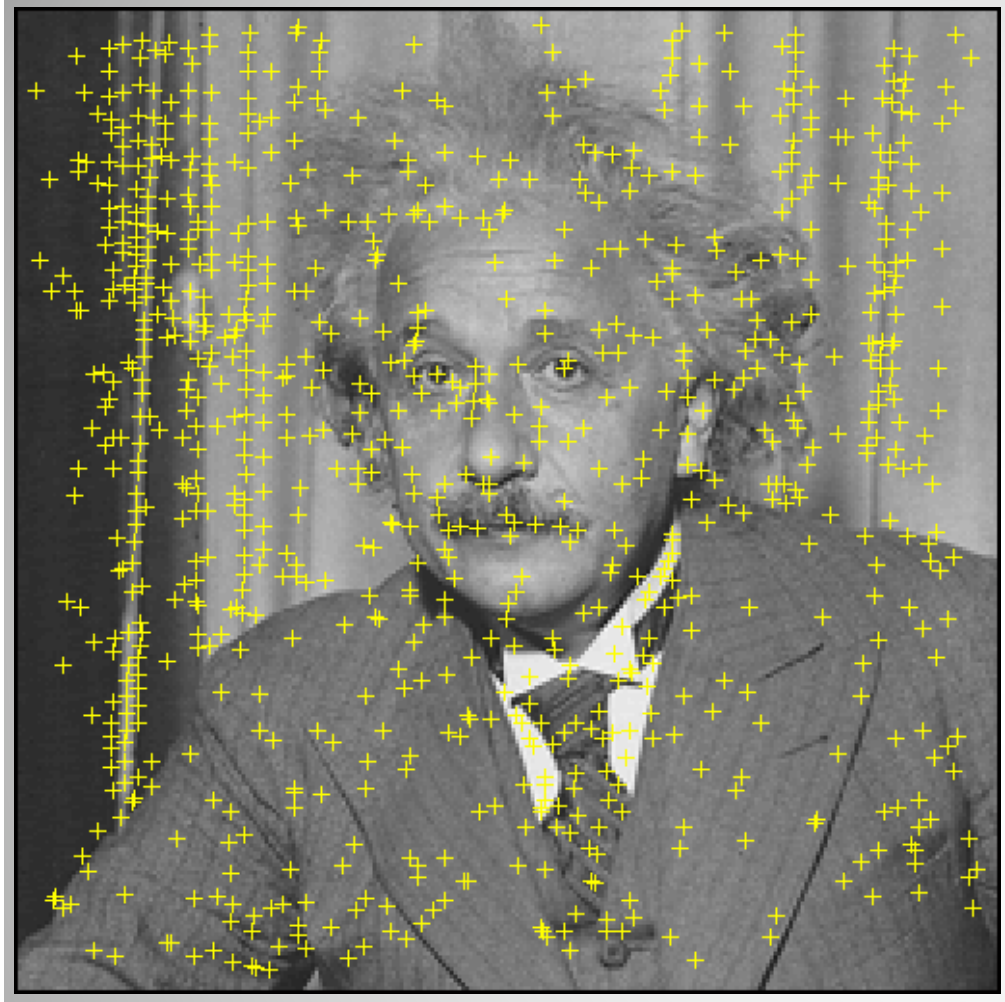
Local extremas



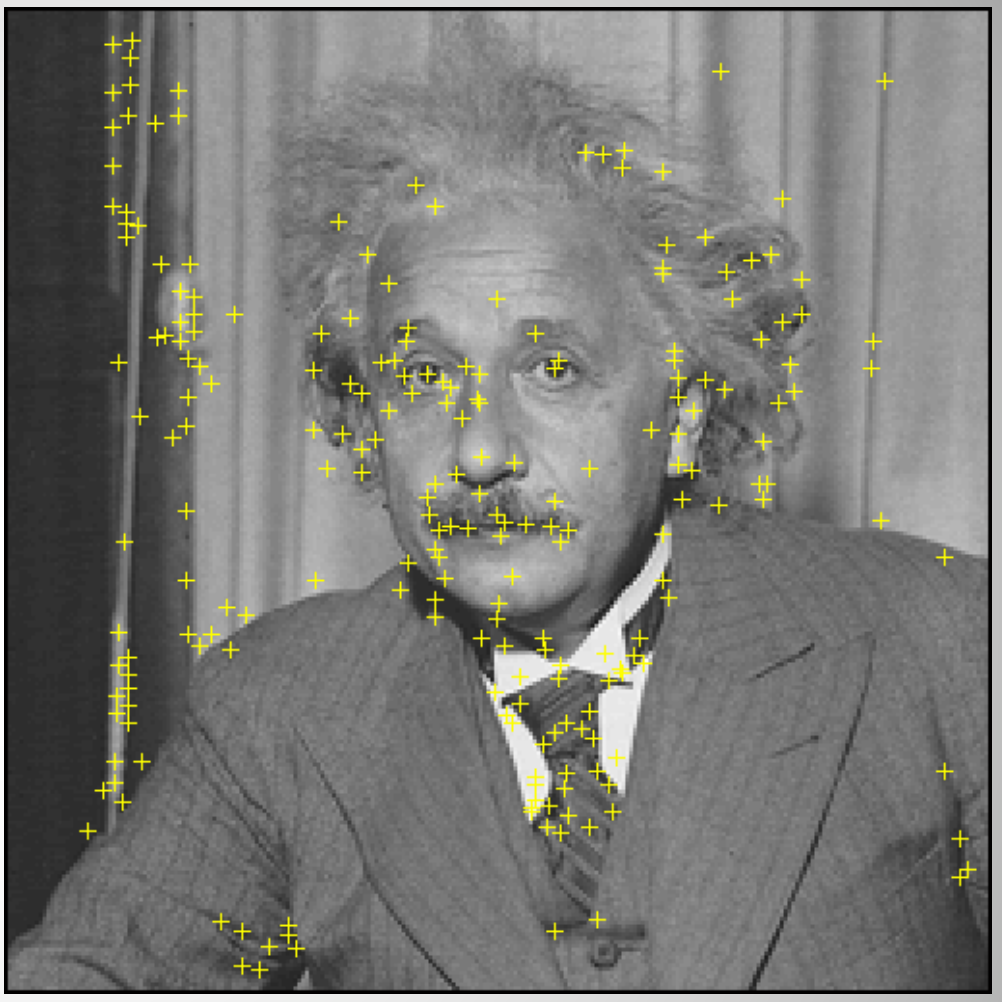
Local extremas



Remove low contrast features

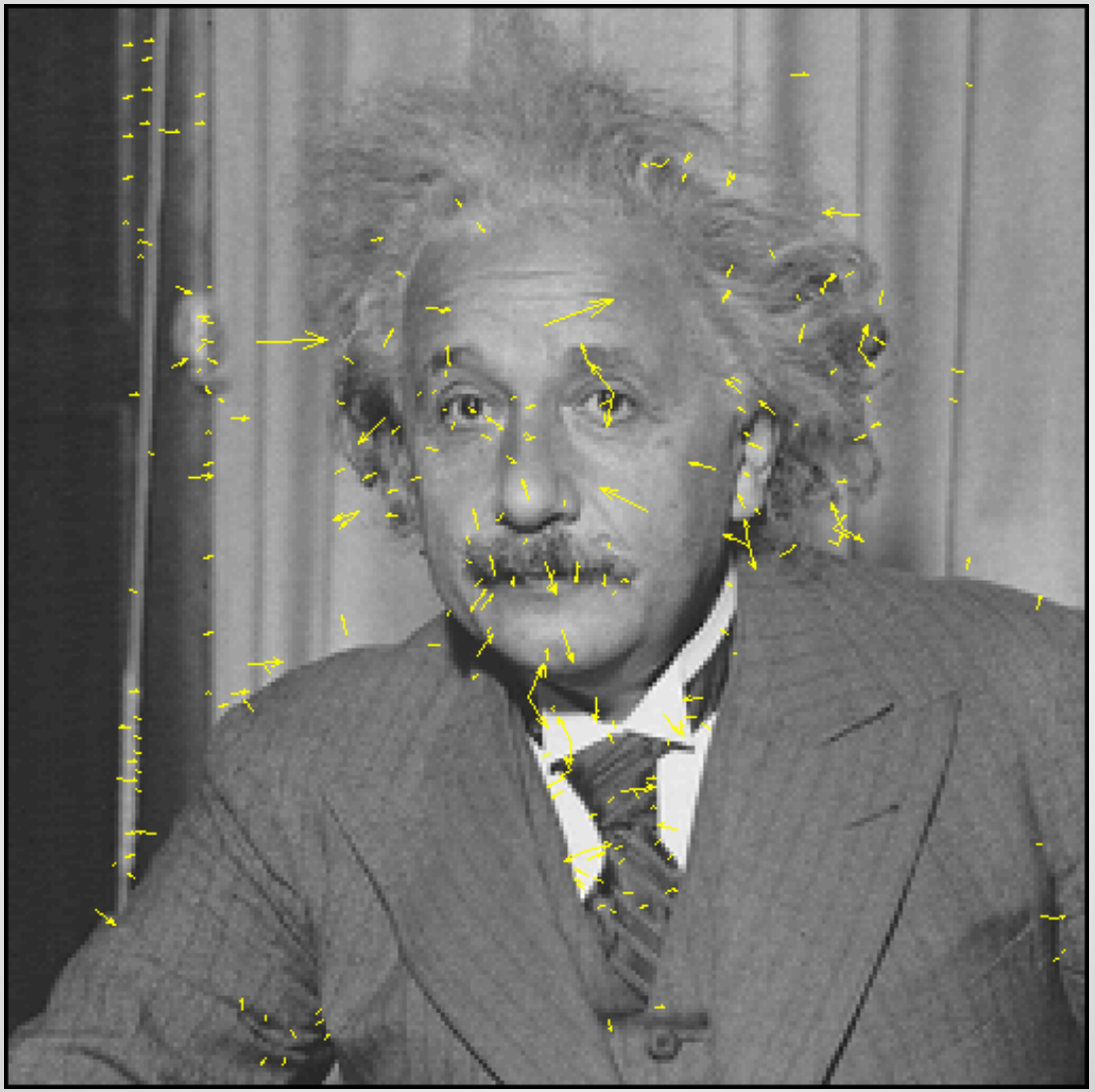


Local extremas



Remove low edges

SIFT Descriptor



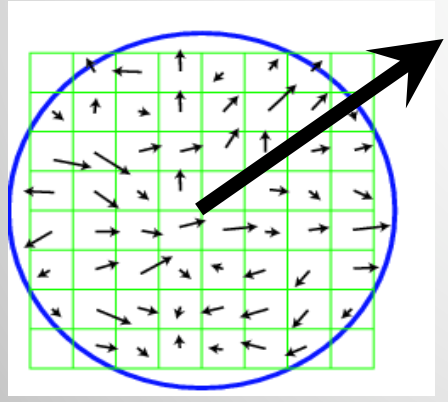
(3) Orientation Assignment

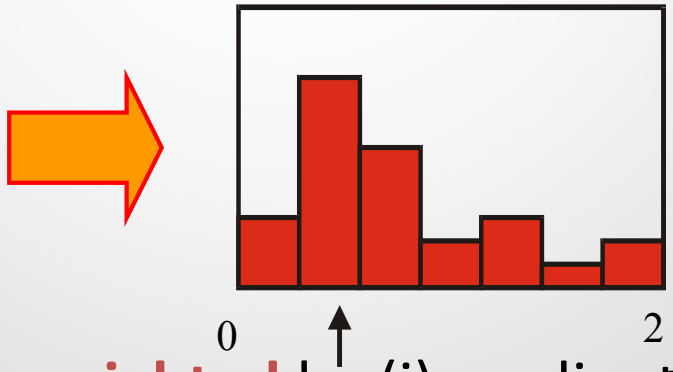
Create histogram of gradient directions, within a region around the keyPoint, at selected scale:

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

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

$$\theta(x,y) = a\tan 2((L(x,y+1) - L(x,y-1)) / (L(x+1,y) - L(x-1,y)))$$

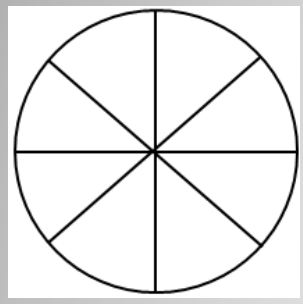


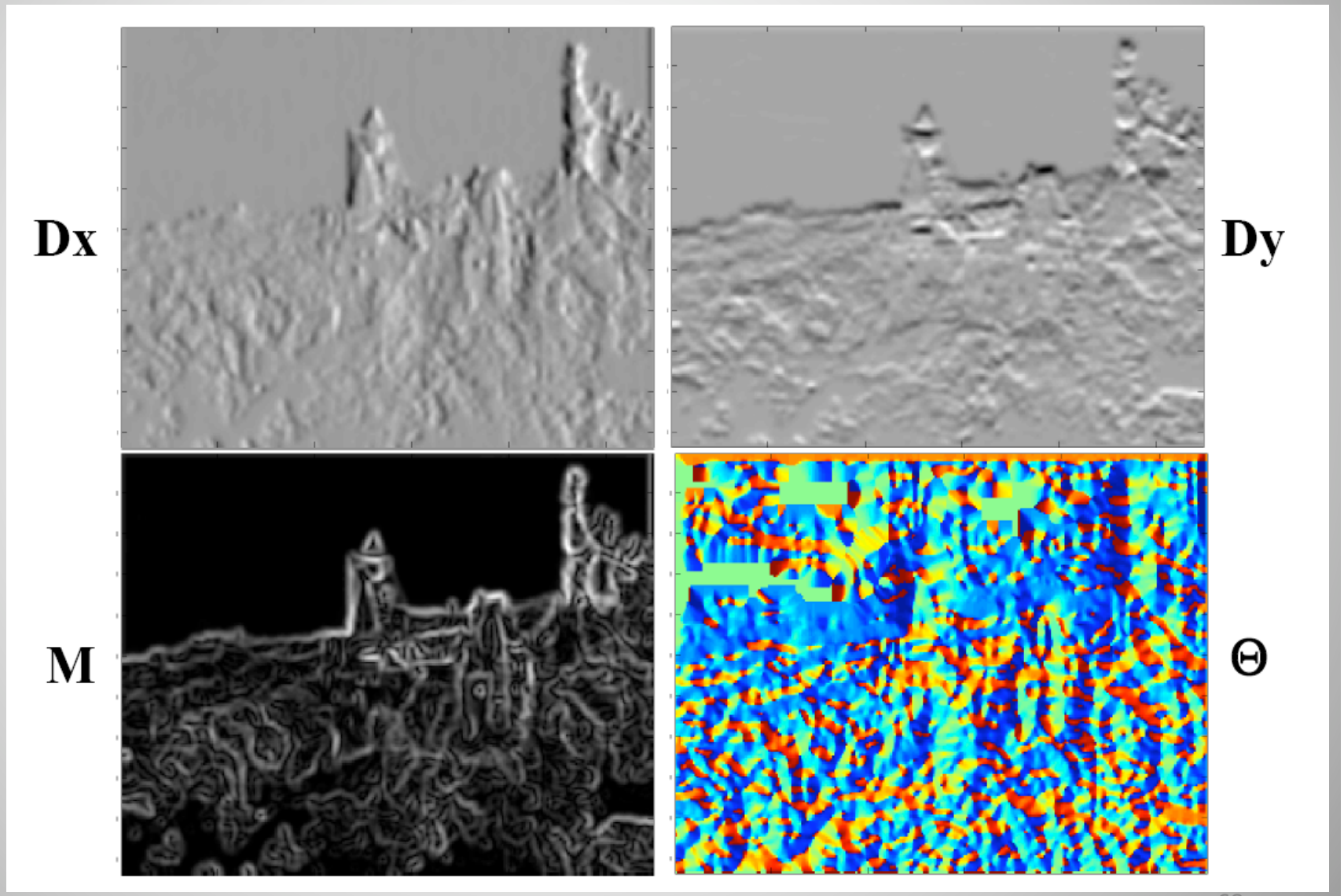


36 bins (i.e., 10^o per bin)

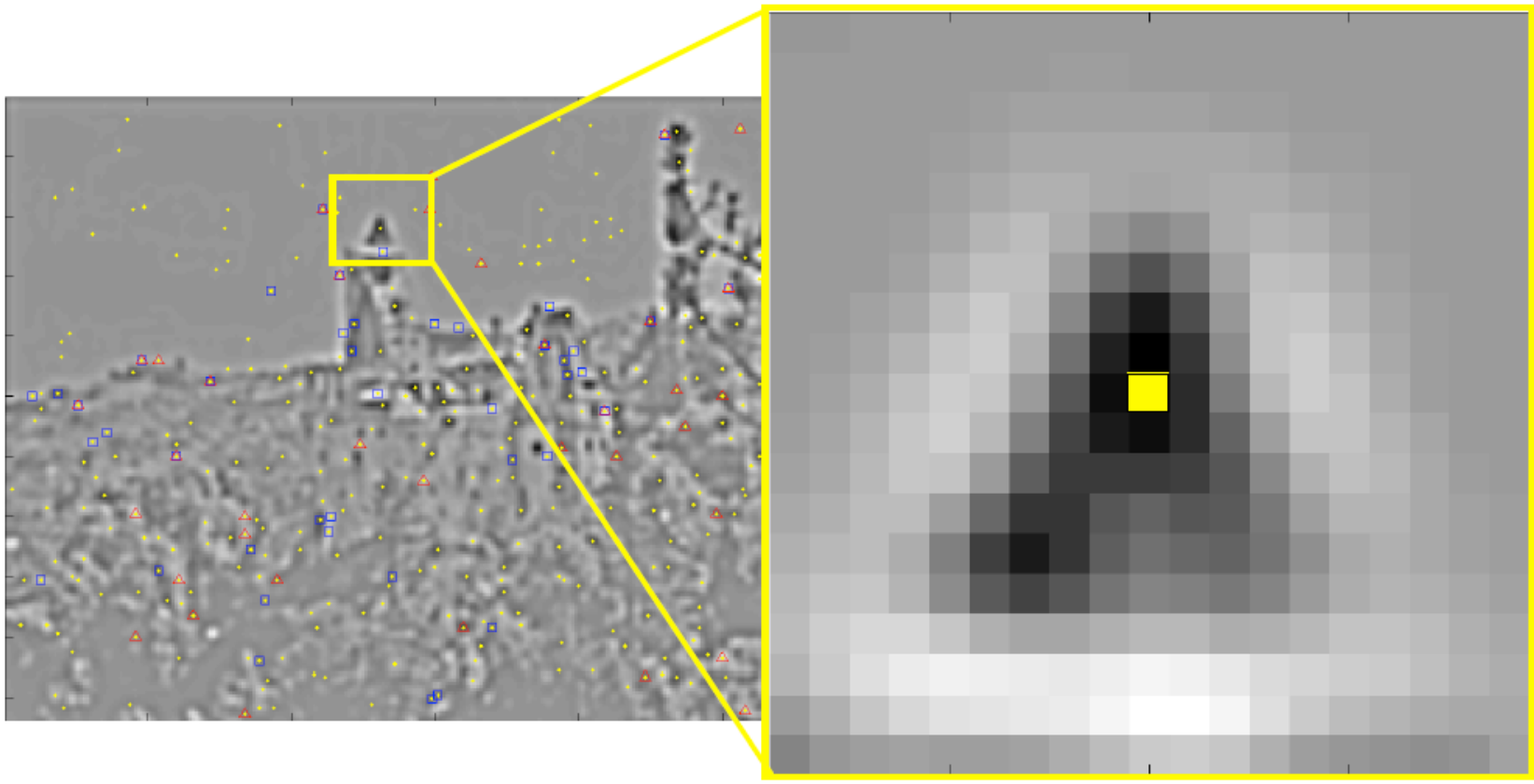
- Histogram entries are **weighted** by (i) gradient magnitude and (ii) a Gaussian function with σ equal to 1.5 times the scale of the keypoint.

(3) Orientation Assignment



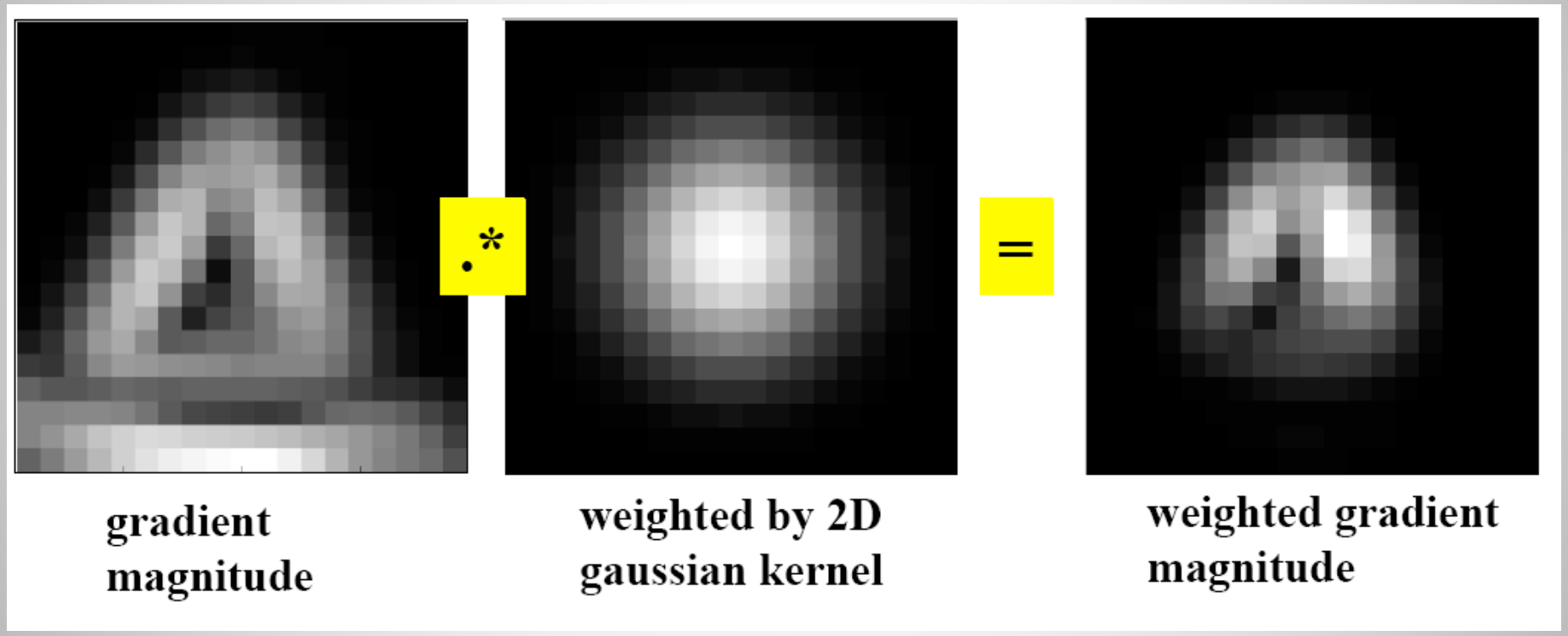


(3) Orientation Assignment



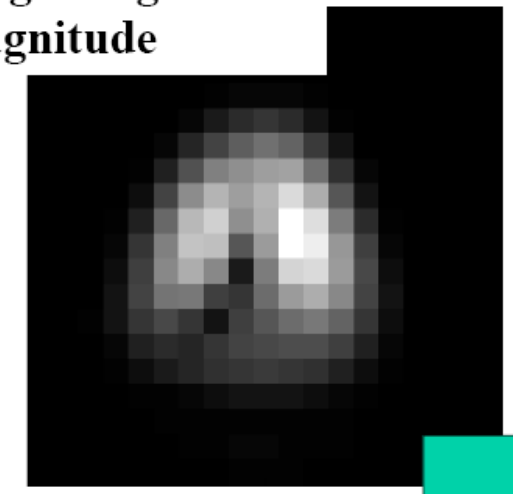
- **Keypoint location = extrema location**
- **Keypoint scale is scale of the DOG image**

(3) Orientation Assignment

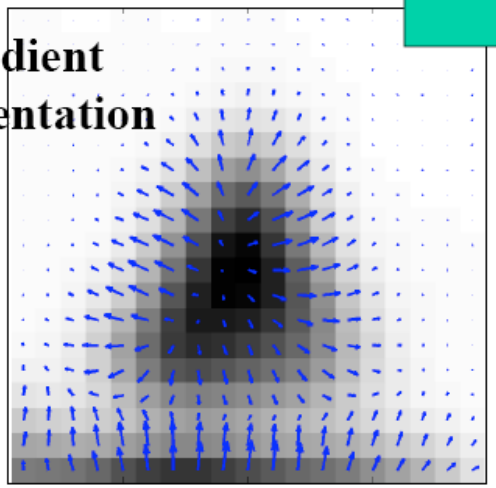


(3) Orientation Assignment

weighted gradient magnitude

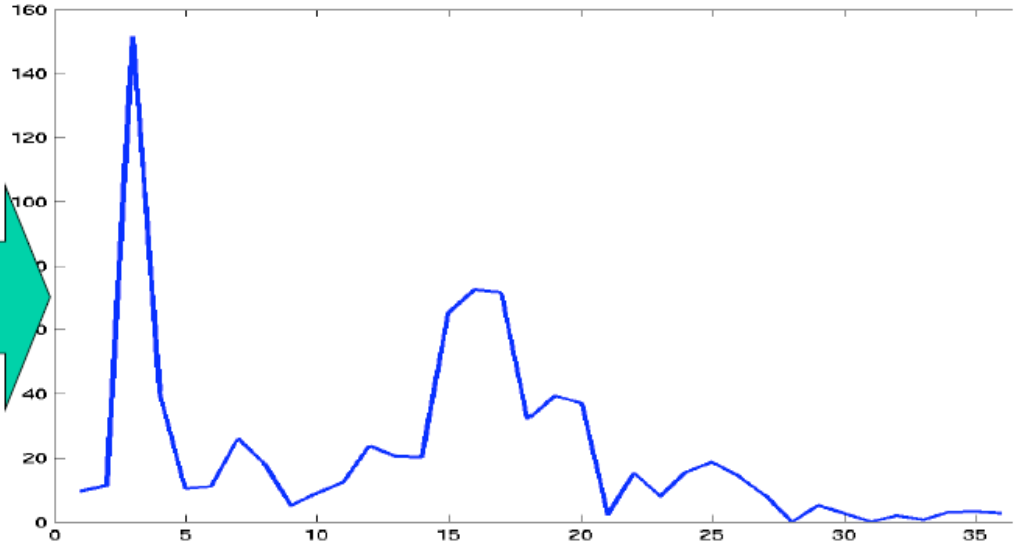


gradient orientation



weighted orientation histogram.

Each bucket contains sum of weighted gradient magnitudes corresponding to angles that fall within that bucket.



36 buckets

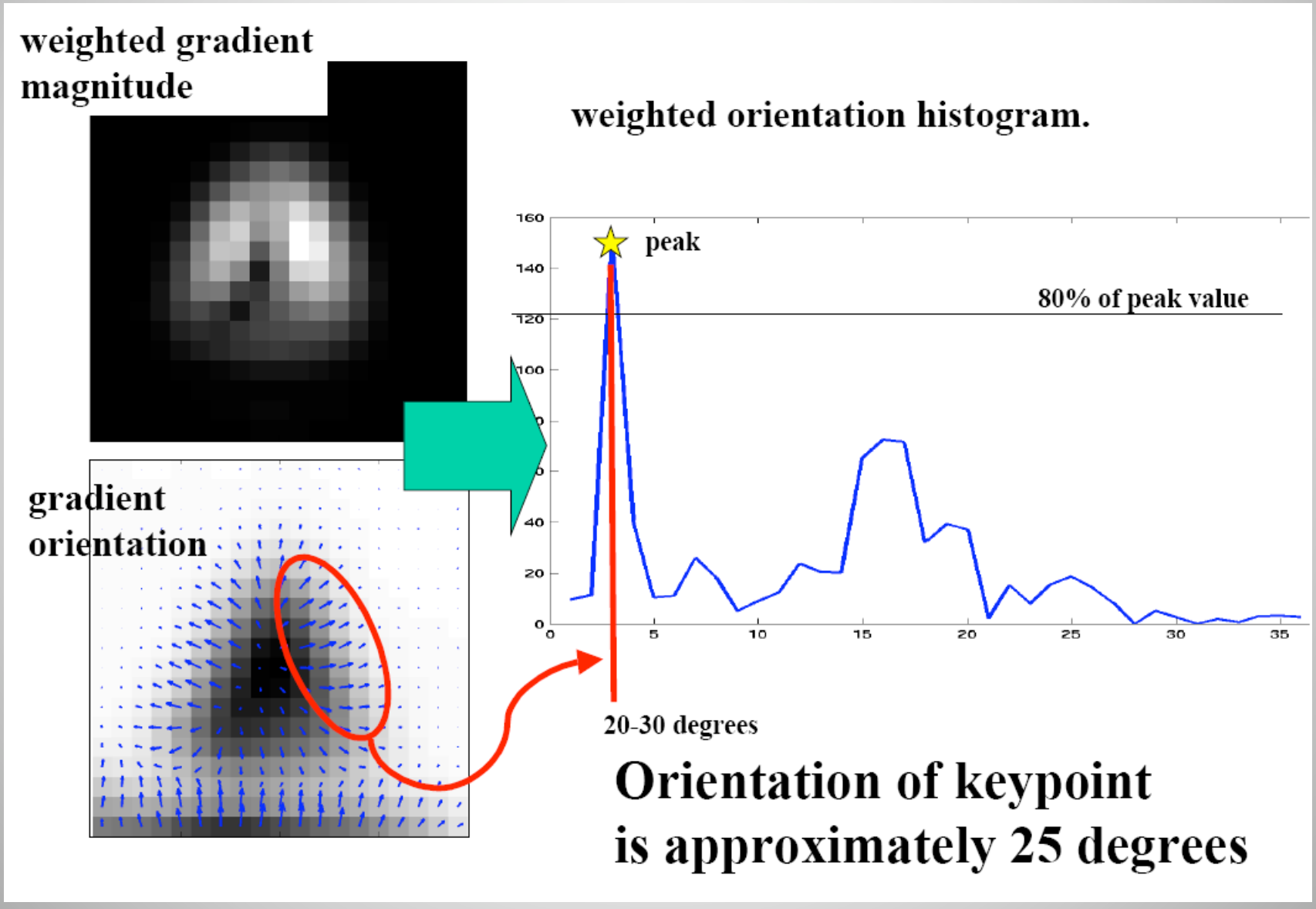
10 degree range of angles in each bucket, i.e.

0 <=ang<10 : bucket 1

10<=ang<20 : bucket 2

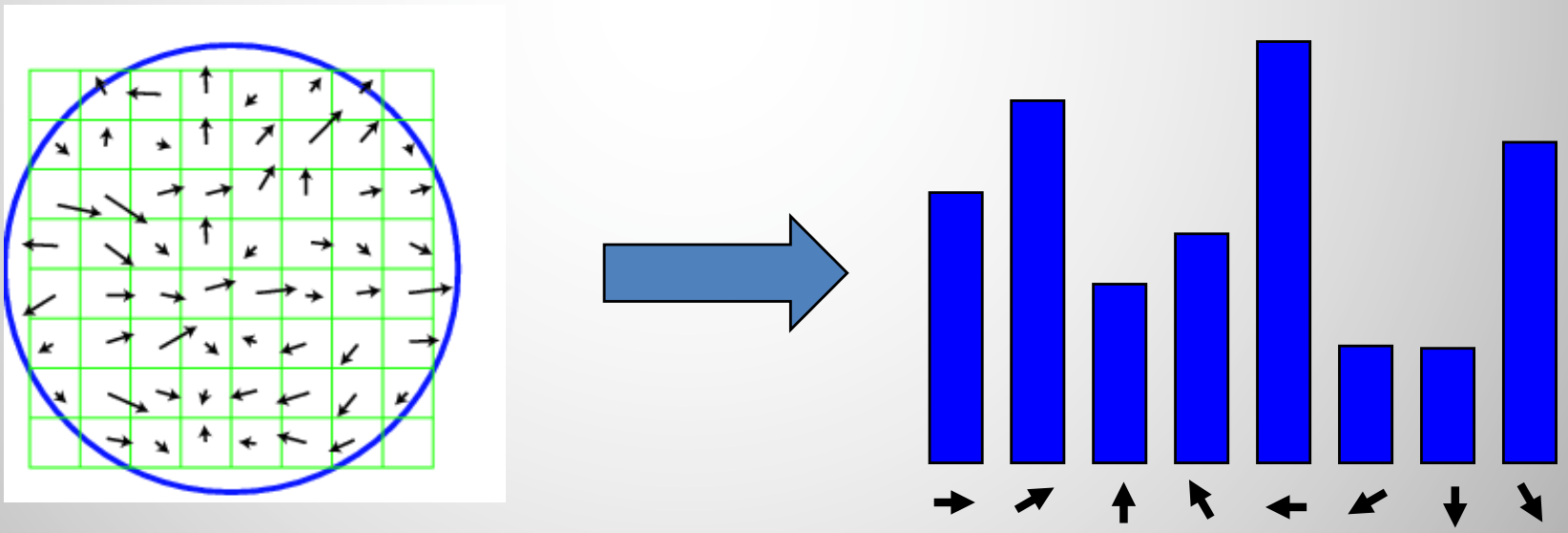
20<=ang<30 : bucket 3 …

(3) Orientation Assignment



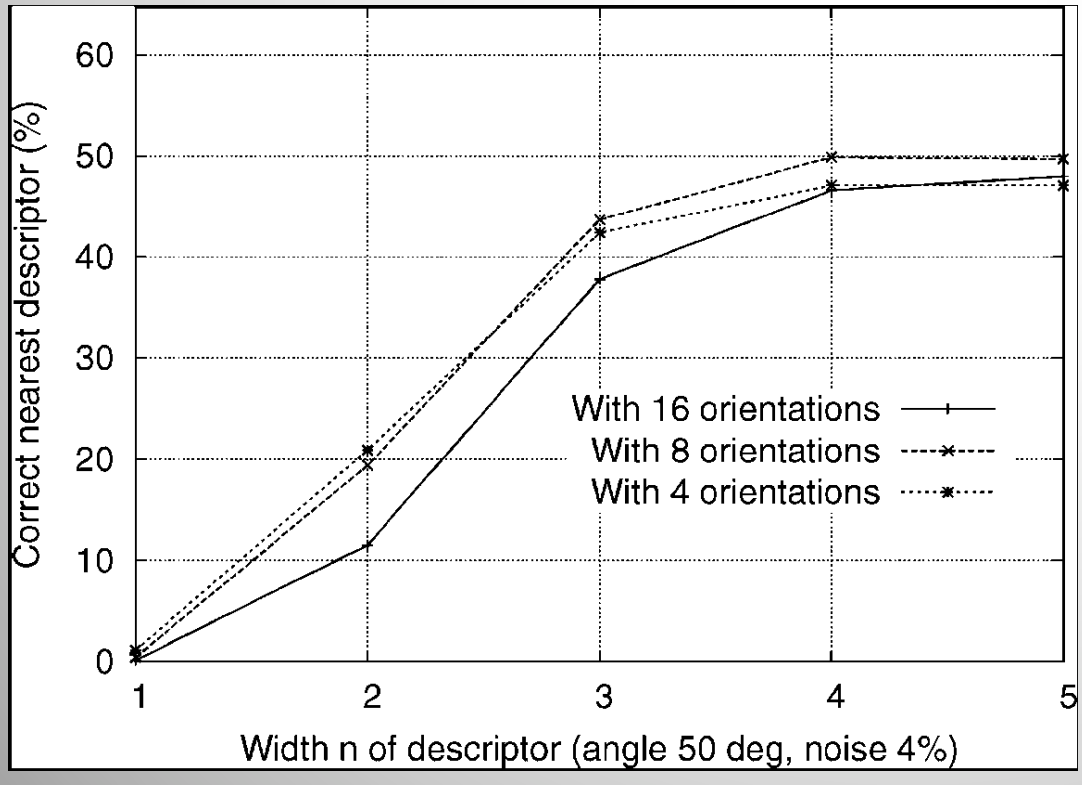
(4) KeyPoint Descriptor

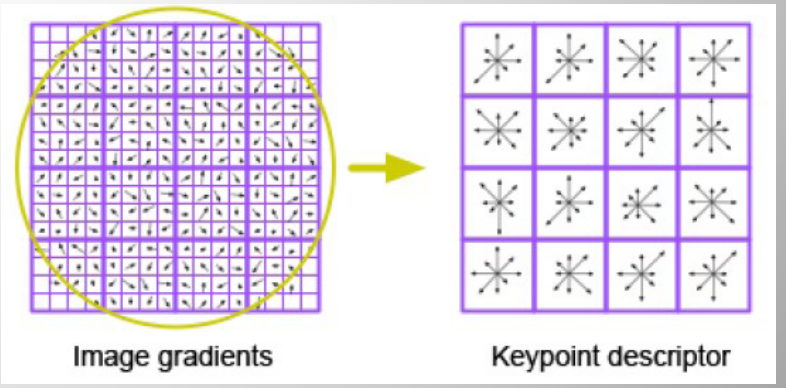
- Partial Voting: distribute histogram entries into adjacent bins (i.e., **additional robustness to shifts)**
  - Each entry is added to all bins, multiplied by a weight of **1-d,** where **d** is the distance from the bin it belongs.



(4) KeyPoint Descriptor

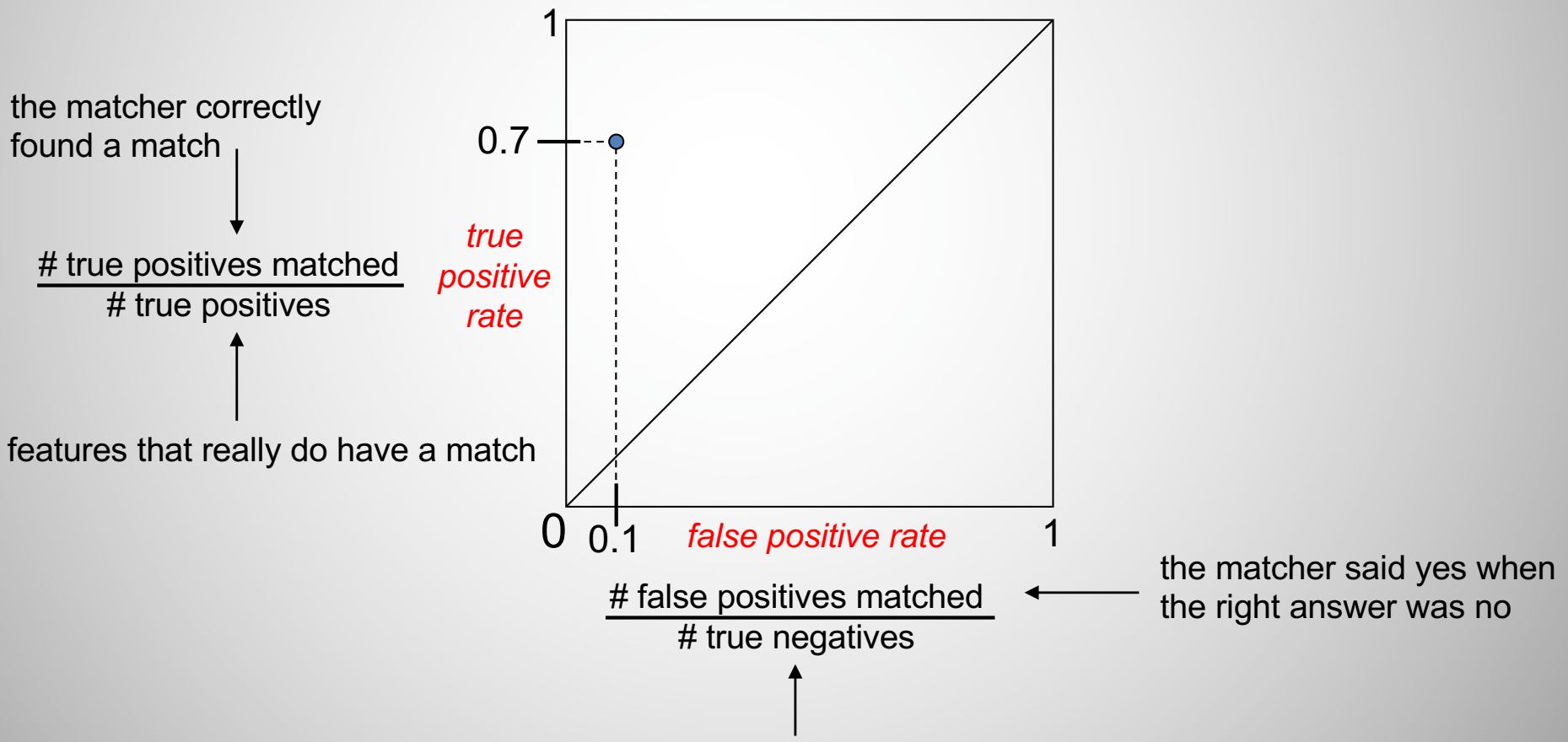
- Descriptor depends on two main parameters:
  - (1) number of orientations
  - n x n array of orientation histograms





Evaluating Results

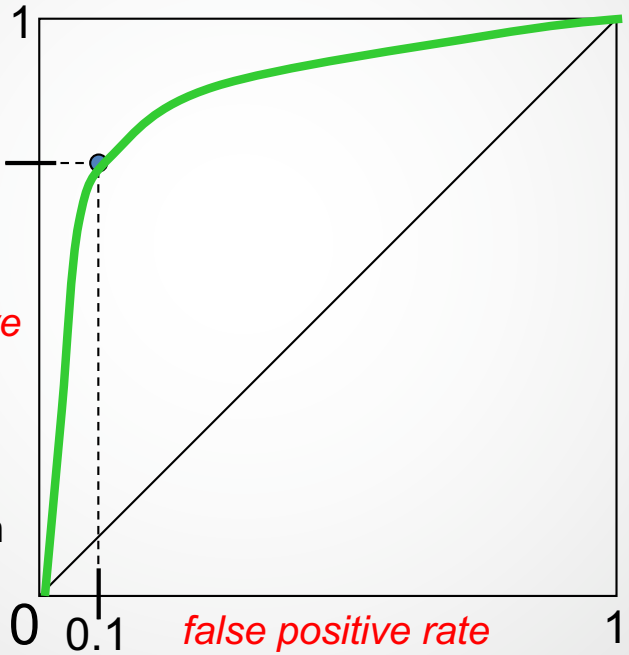
How can we measure the performance of a feature matcher?



Evaluating Results

How can we measure the performance of a feature matcher?

ROC curve **("Receiver Operator Characteristic")**



the matcher correctly found a match

$$\frac{\text{\# true positives matched}}{\text{\# true positives}}$$

features that really do have a match

$$\frac{\text{\# false positives matched}}{\text{\# true negatives}}$$

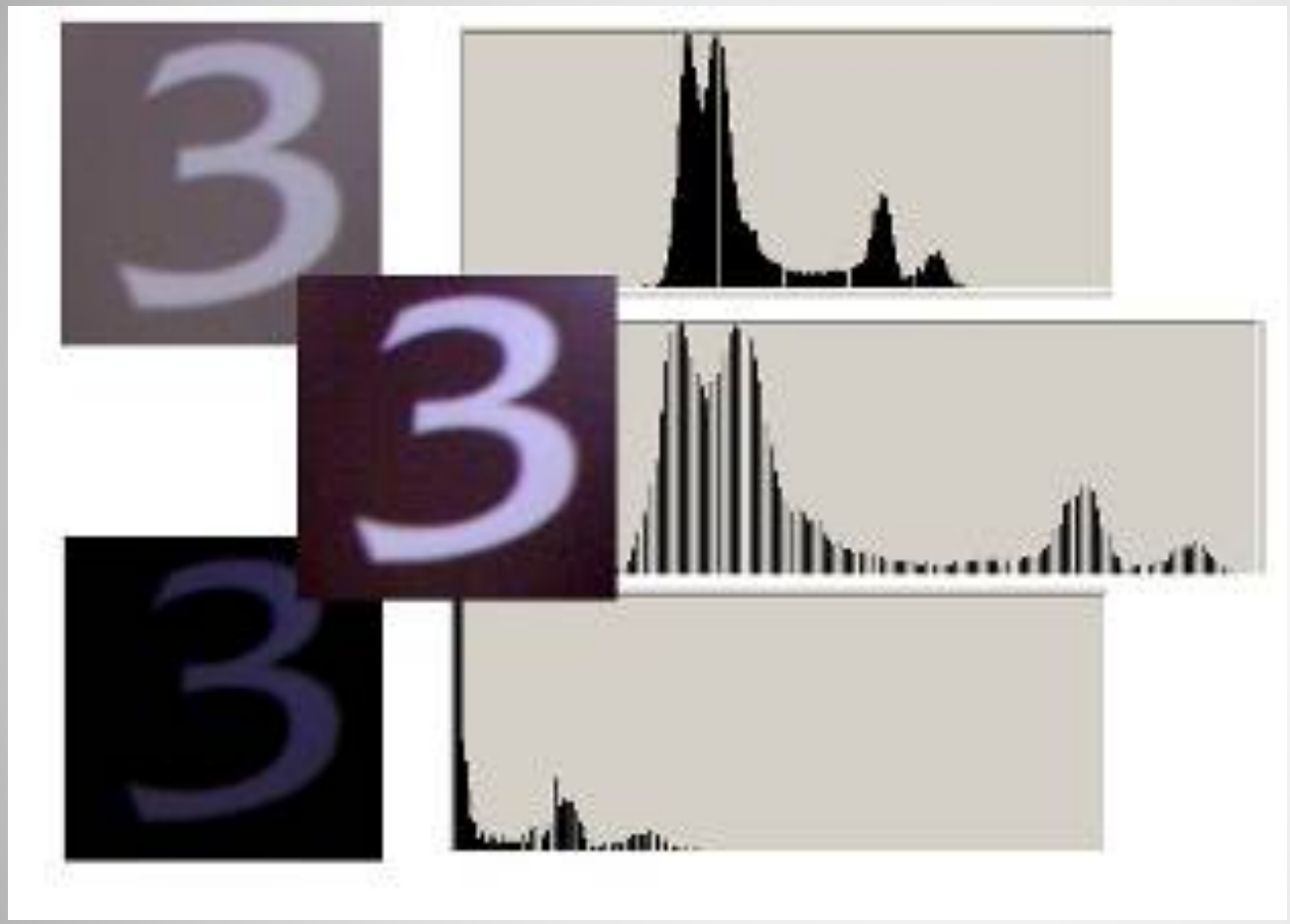
the matcher said yes when the right answer was no

Change of Illumination

- **Change of brightness =>** doesn't effect gradients (difference of pixels value).
- **Change of contrast =>** doesn't effect gradients (up to normalization).
- **Saturation (non-linear change of illumination) =>** affects magnitudes much more than orientation.

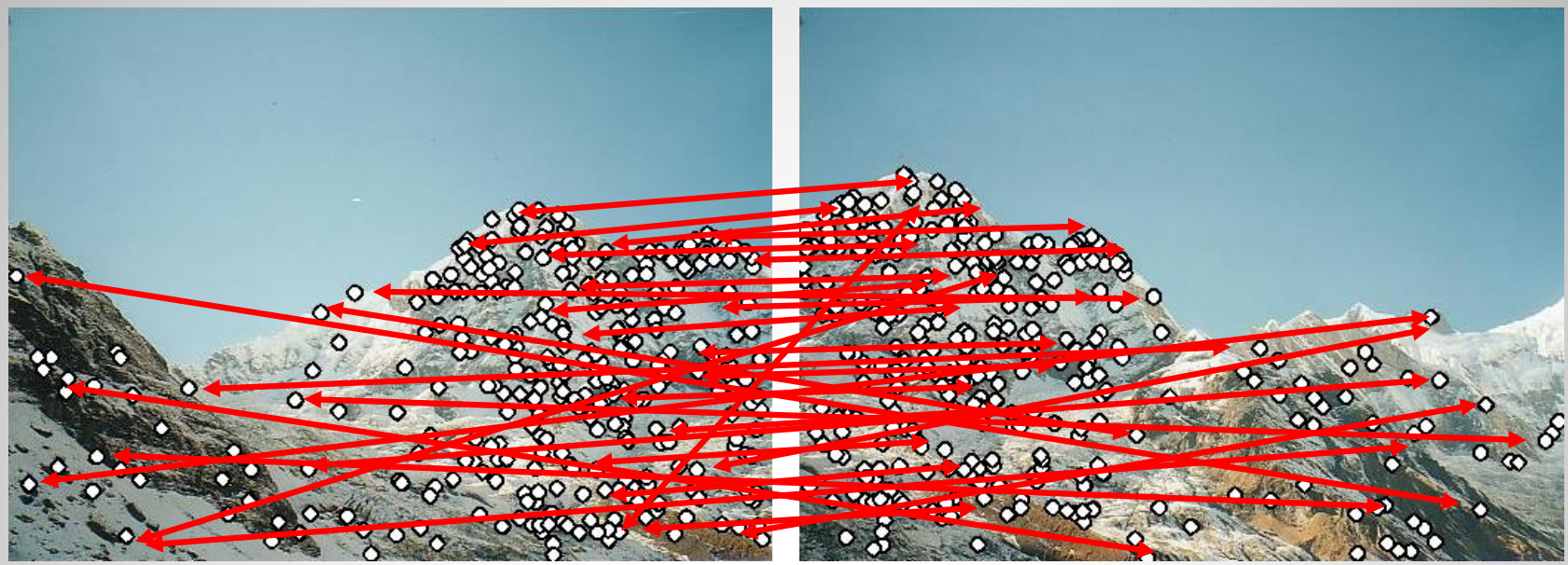
  => Threshold gradient magnitudes to 0.2 and renormalize.

Illumination invariance?

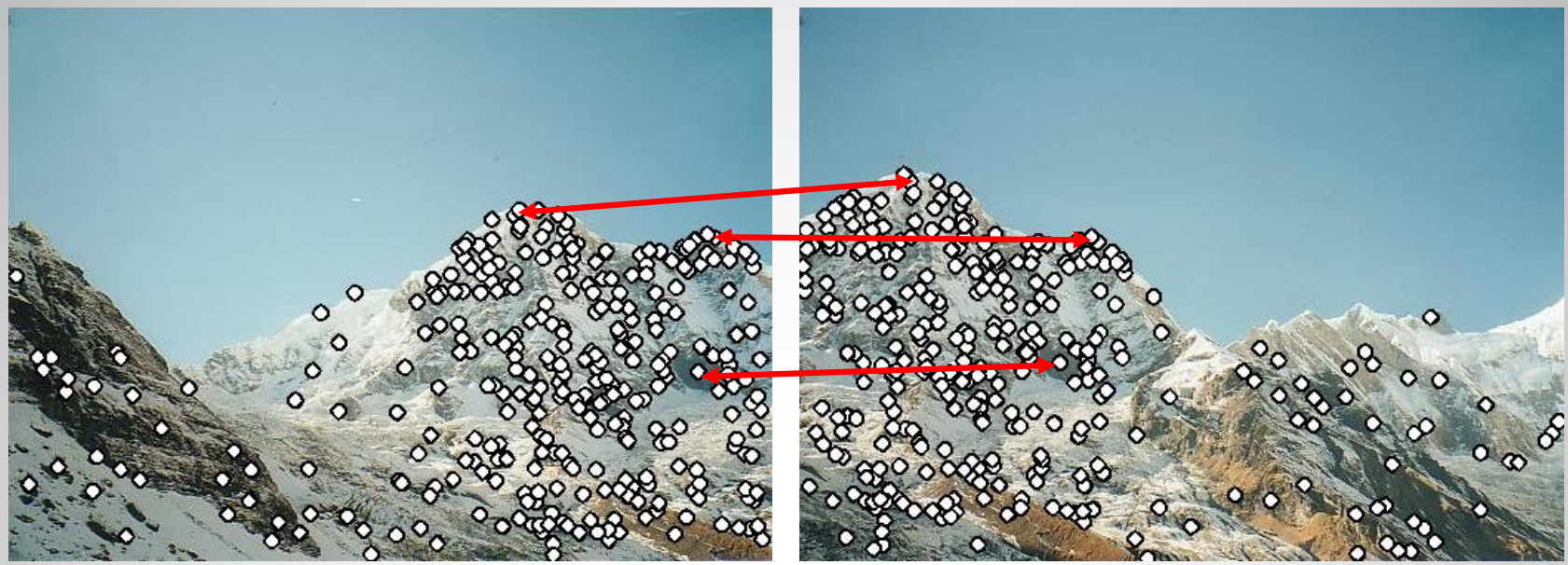


1. Before you start, you can normalize the images.
2. Difference based metrics can be used (Haar, SIFT,…)

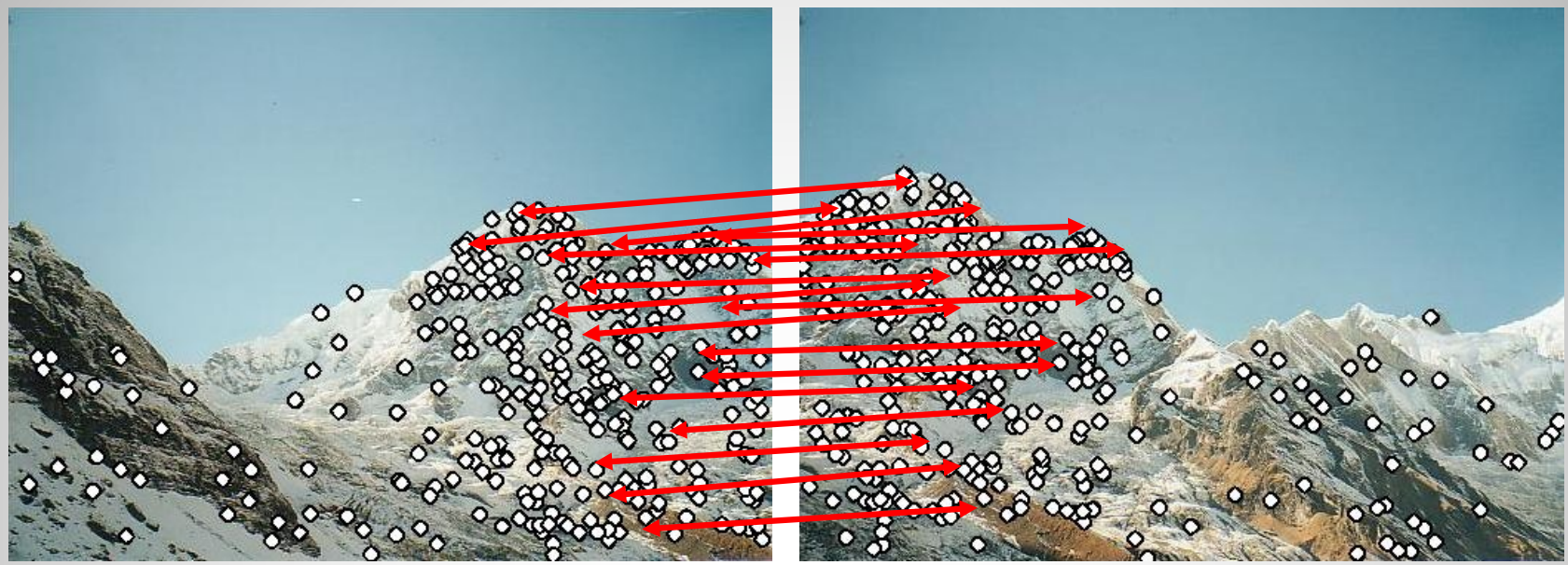
- Extract features



- Extract features
- Compute *putative matches*



- Extract features
- Compute *putative matches*
- Loop:
  - *Hypothesize* transformation *T* (small group of putative matches that are related by *T*)



- Extract features
- Compute *putative matches*
- Loop:
  - *Hypothesize* transformation *T* (small group of putative matches that are related by *T*)
  - *Verify* transformation (search for other matches consistent with *T*)

- Extract features
- Compute *putative matches*
- Loop:
  - *Hypothesize* transformation *T* (small group of putative matches that are related by *T*)
  - *Verify* transformation (search for other matches consistent with *T*)

Object Recognition

- **For training images:**
  - Extracting keypoints by SIFT.
  - Creating descriptors database.
- **For query images:**
  - Extracting keypoints by SIFT.
  - For each descriptor - finding nearest neighbor in DB.
  - Finding cluster of at-least 3 keypoints.
  - Performing detailed geometric fit check for each cluster.

Image Registration