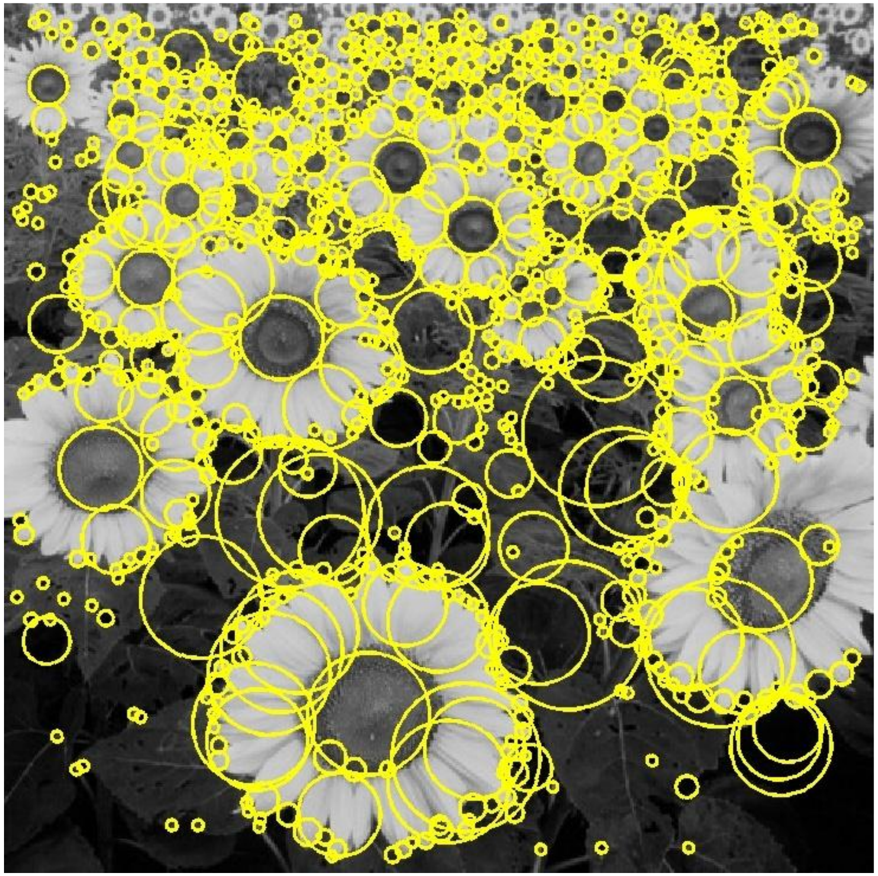
Feature Detectors/Descriptors

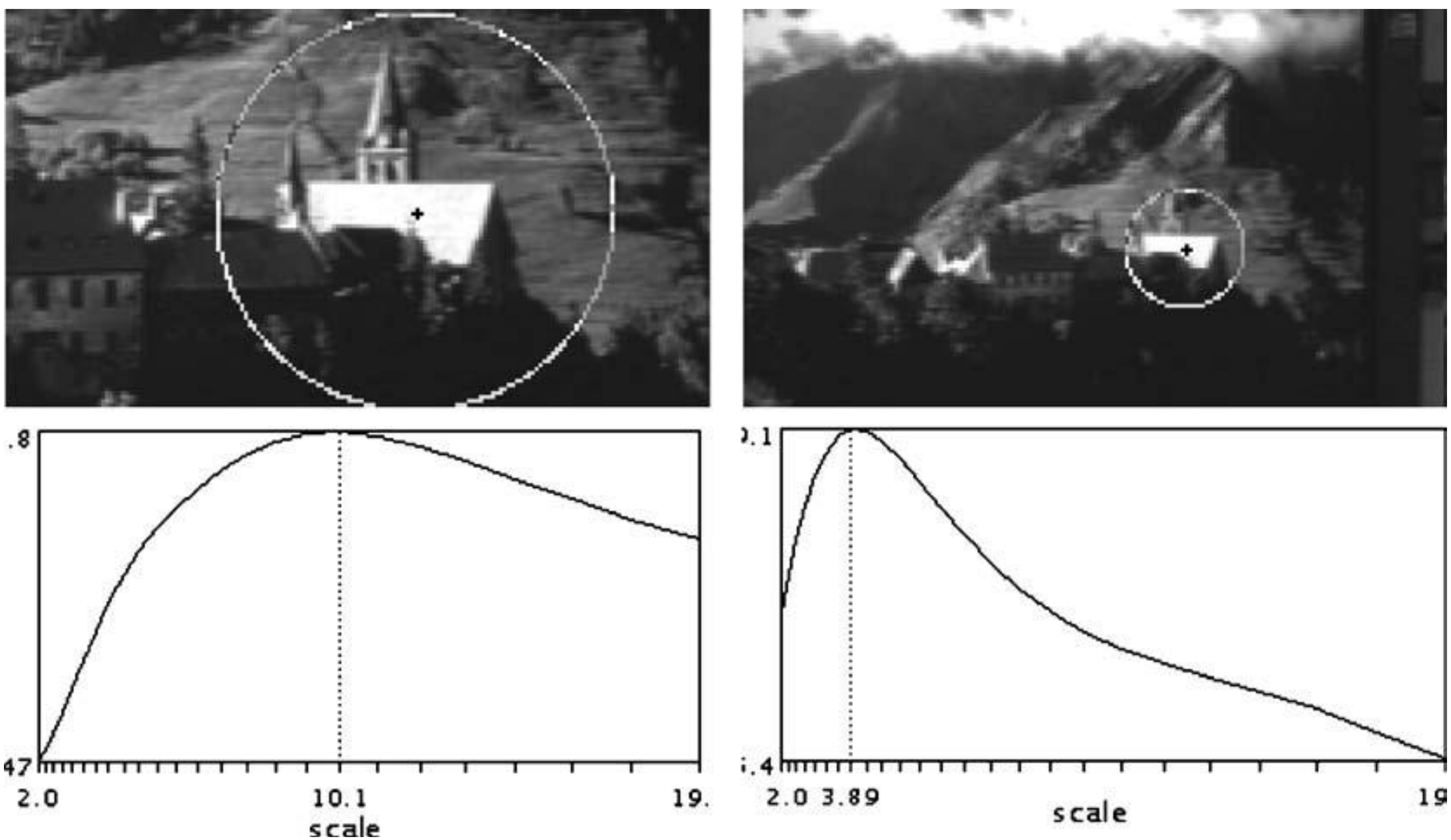
Book: Szeliski 4.1.2 , Forsyth 4.5-4.7, 5.4-5.5,

Blob detection



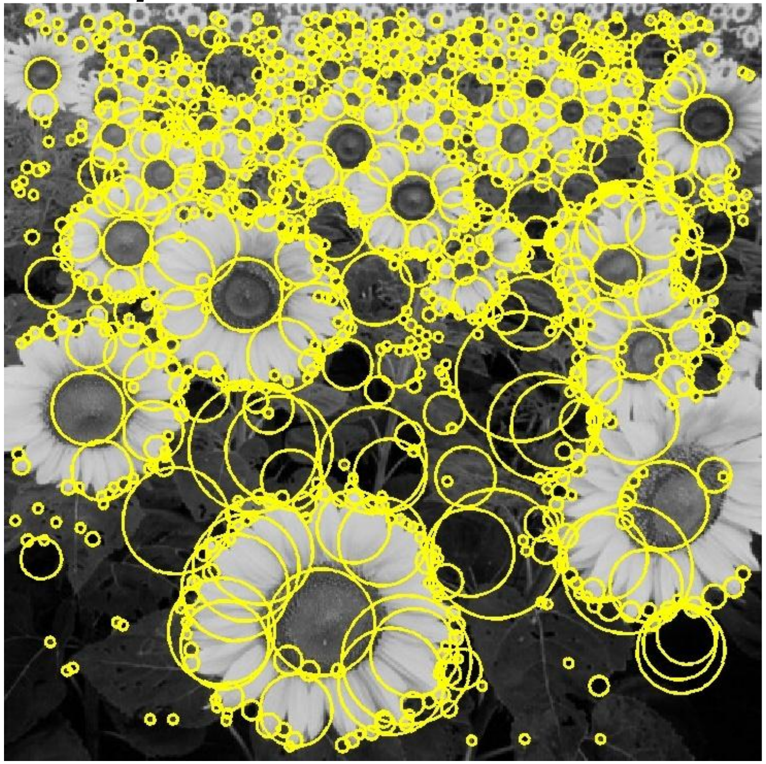
Feature detection with scale selection

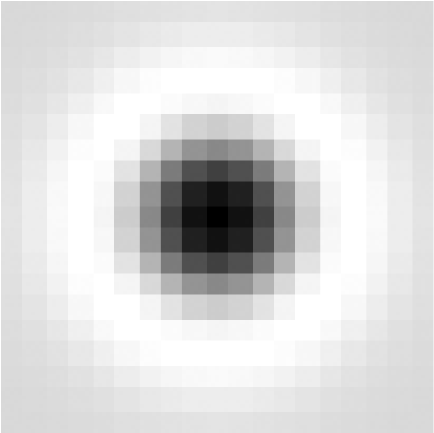
- We want to extract features with characteristic scale that is *covariant* with the image transformation



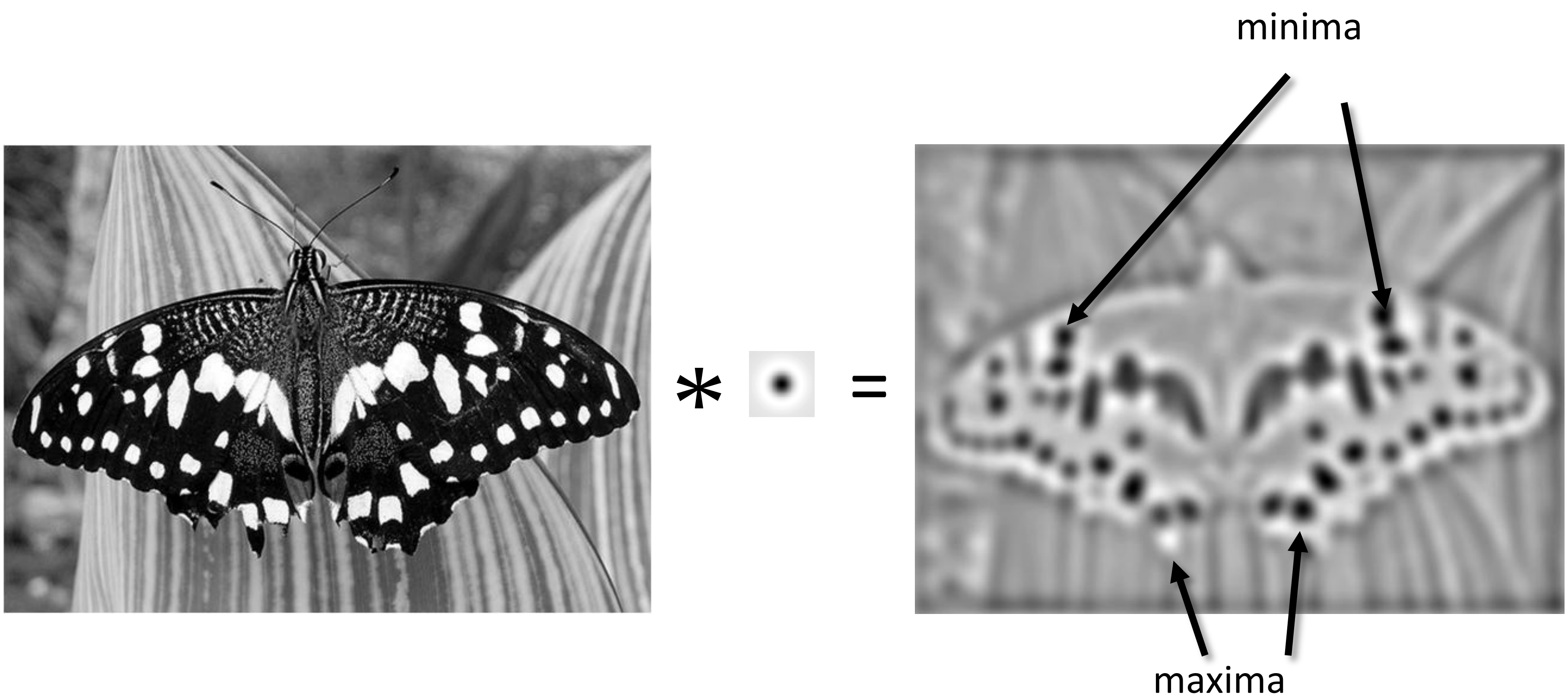
Blob detection: Basic idea

- To detect blobs, convolve the image with a "blob filter" at multiple scales and look for extrema of filter response in the resulting *scale space*





Blob detection: Basic idea

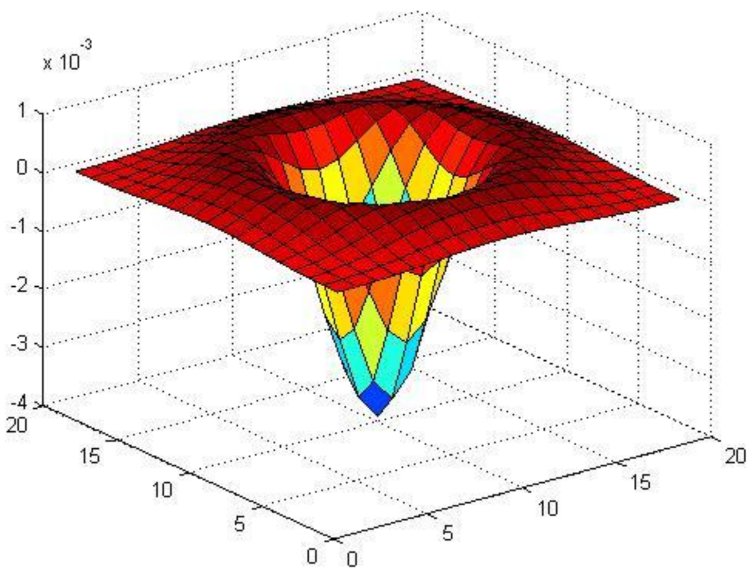


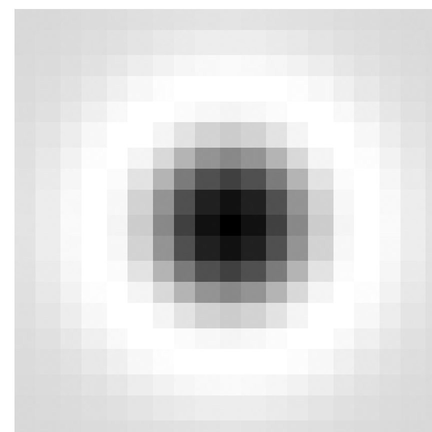
- Find maxima *and minima* of blob filter response in space *and scale*

Source: N. Snavely

Blob filter

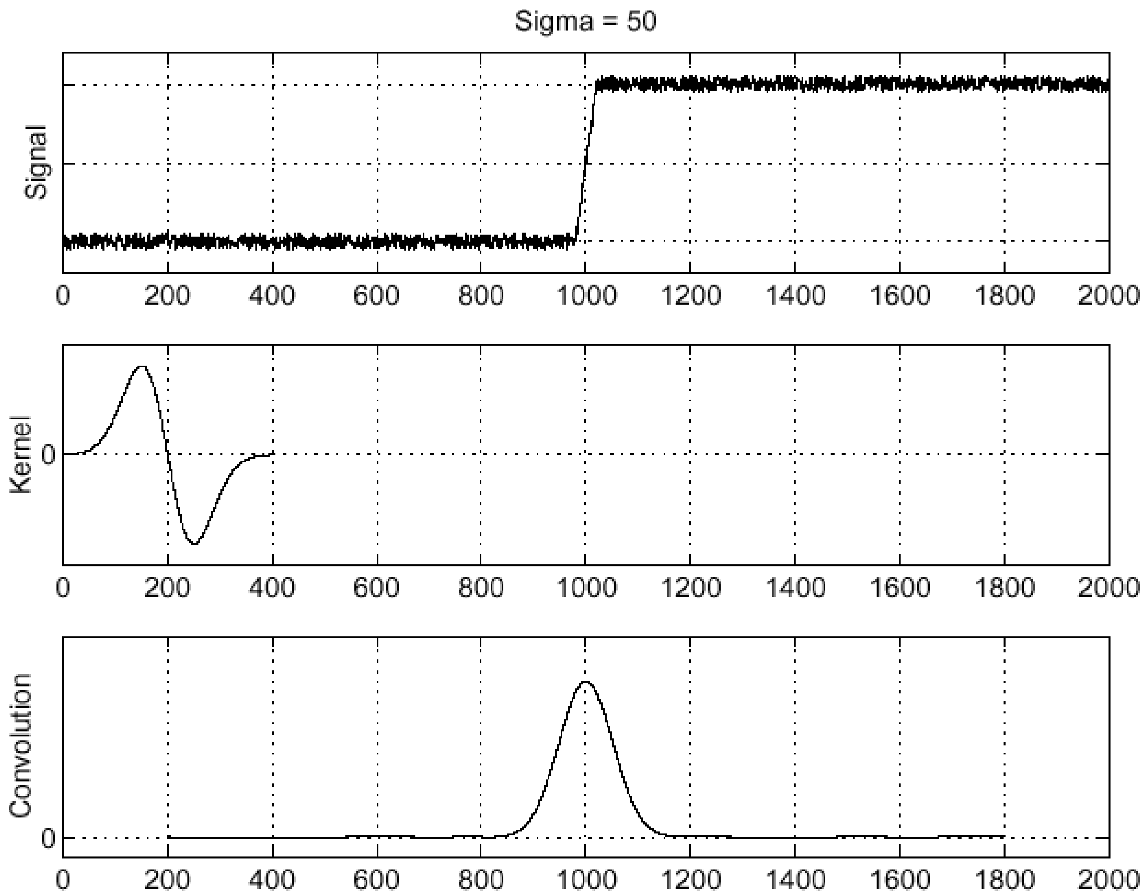
- Laplacian of Gaussian: Circularly symmetric operator for blob detection in 2D





$$\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}$$

Recall: Edge detection



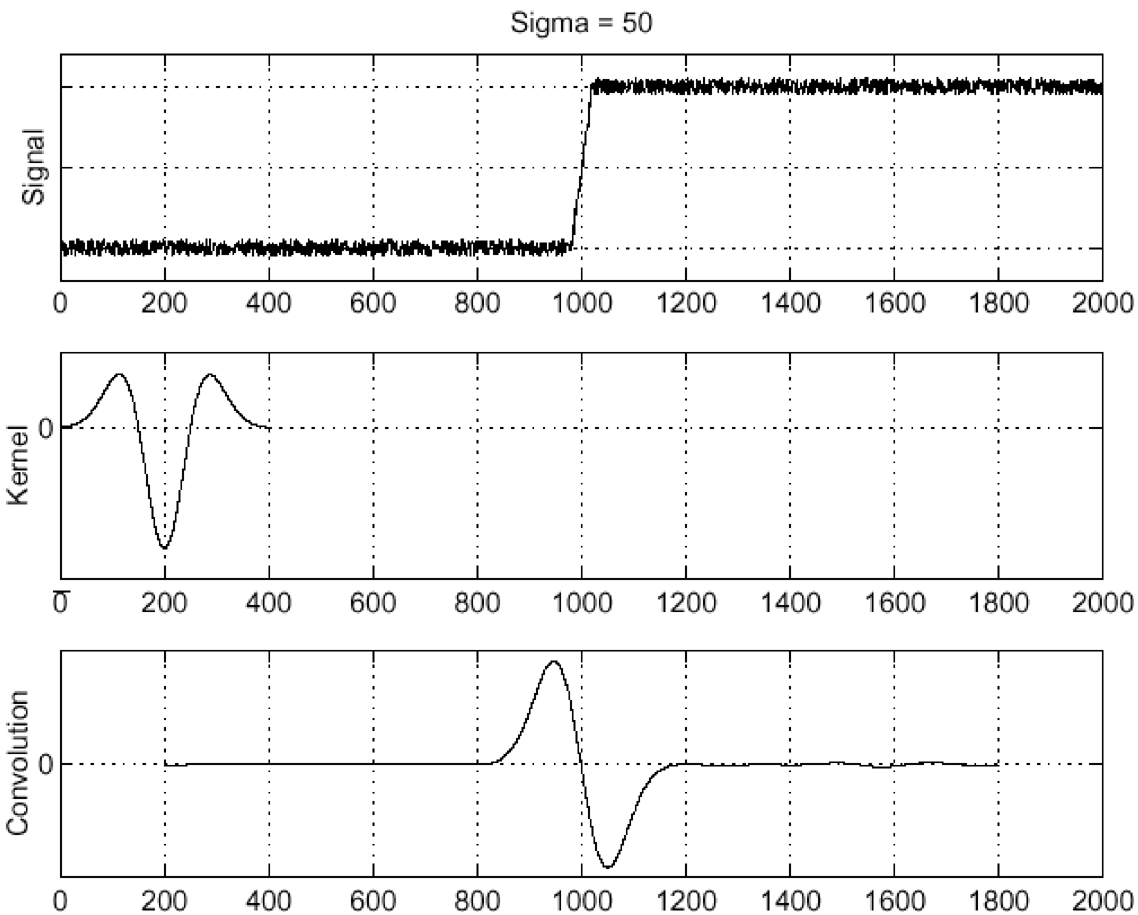
f — Edge

$\frac{d}{dx} g$ — Derivative of Gaussian

$f * \frac{d}{dx} g$ — Edge = maximum of derivative

Source: S. Seitz

Edge detection, Take 2



$$f$$

Edge

$$\frac{d^2}{dx^2} g$$

Second derivative of Gaussian (Laplacian)

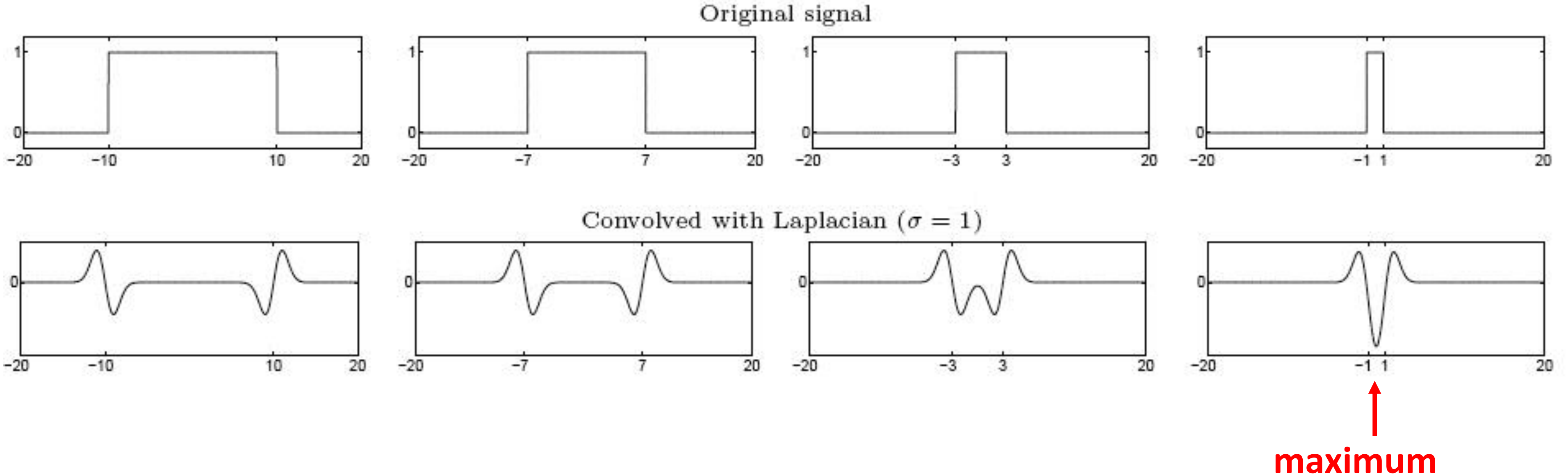
$$f * \frac{d^2}{dx^2} g$$

Edge = zero crossing of second derivative

Source: S. Seitz

From edges to blobs

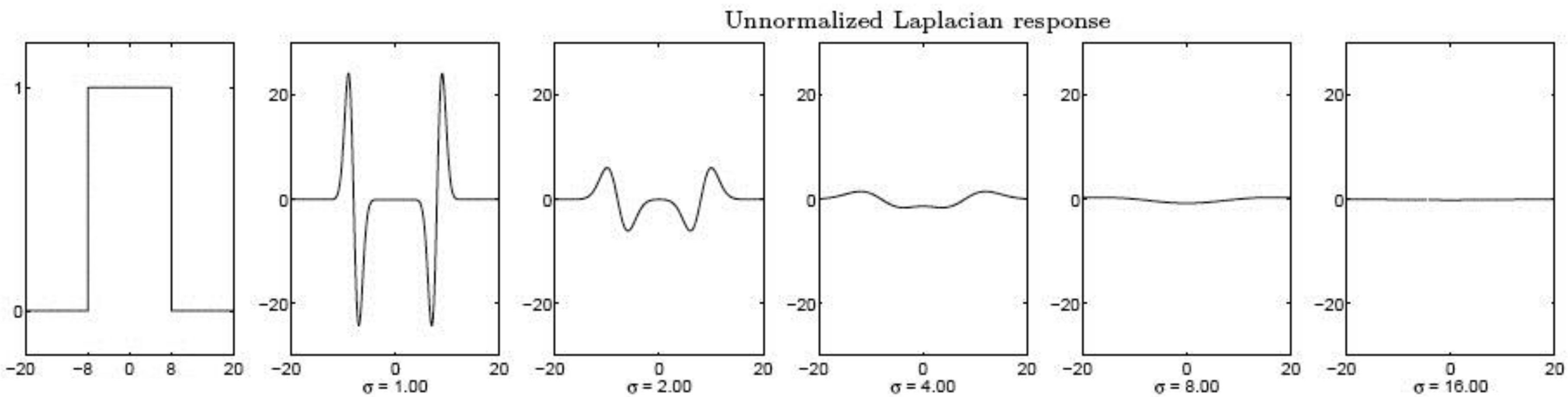
- Edge = ripple
- Blob = superposition of two ripples



Spatial selection: the magnitude of the Laplacian response will achieve a maximum at the center of the blob, provided the scale of the Laplacian is "matched" to the scale of the blob

Scale selection

- We want to find the characteristic scale of the blob by convolving it with Laplacians at several scales and looking for the maximum response
- However, Laplacian response decays as scale increases:



original signal (radius=8)

increasing σ

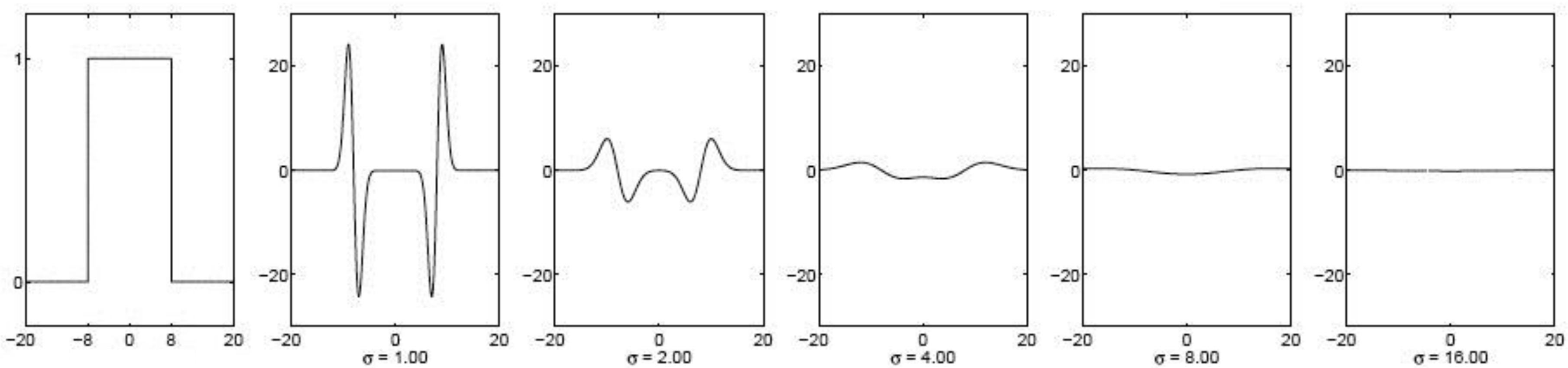
Scale normalization

- The response of a derivative of Gaussian filter to a perfect step edge decreases as σ increases
- To keep response the same (scale-invariant), must multiply Gaussian derivative by σ
- Laplacian is the second Gaussian derivative, so it must be multiplied by σ^2

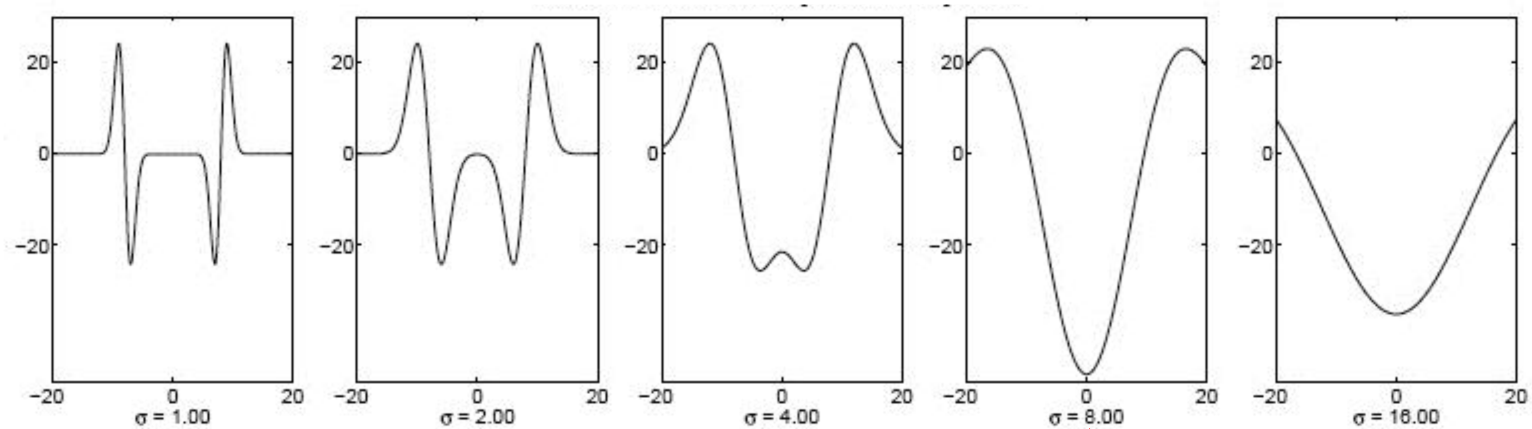
Effect of scale normalization

Original signal

Unnormalized Laplacian response



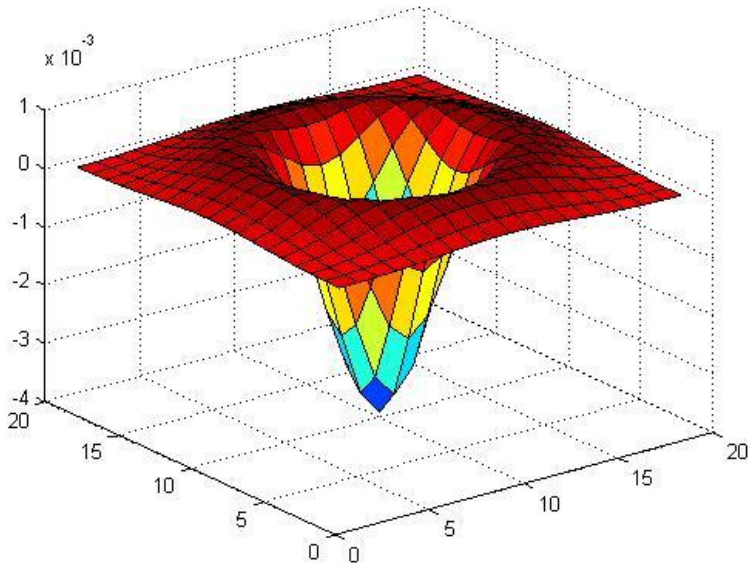
Scale-normalized Laplacian response

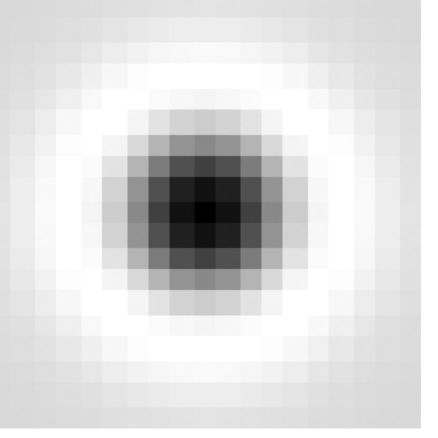


maximum

Blob detection in 2D

- Laplacian of Gaussian: Circularly symmetric operator for blob detection in 2D



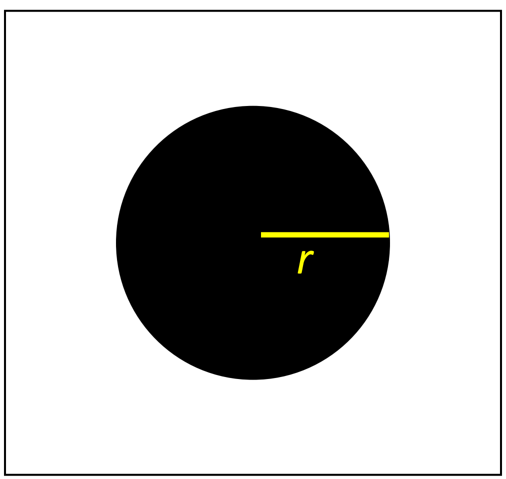


Scale-normalized:

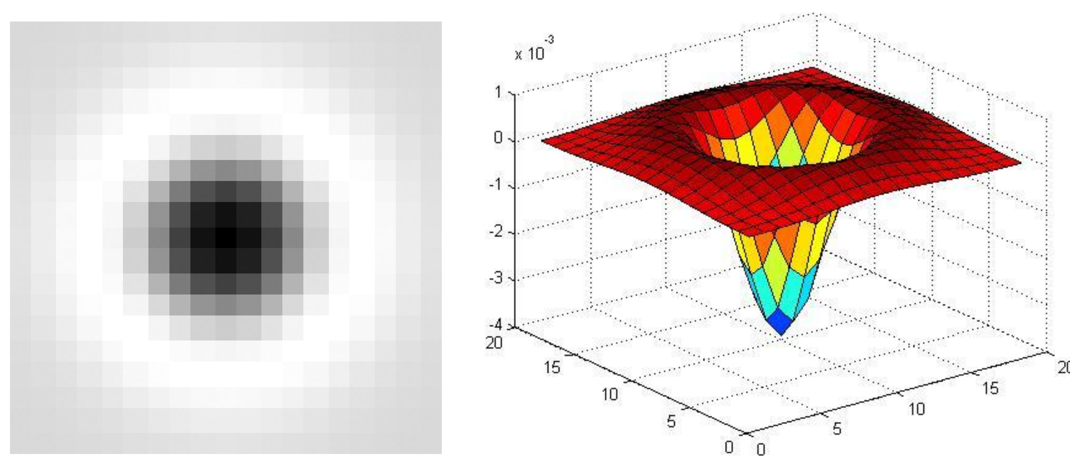
$$\nabla^2_{\text{norm}} g = \sigma^2 \left(\frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} \right)$$

Scale selection

- At what scale does the Laplacian achieve a maximum response to a binary circle of radius r?



image



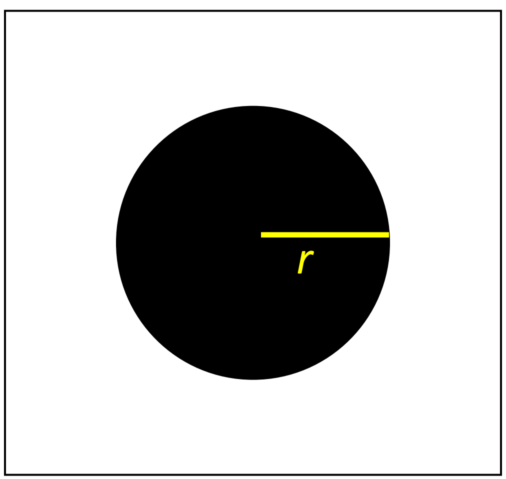
Laplacian

Scale selection

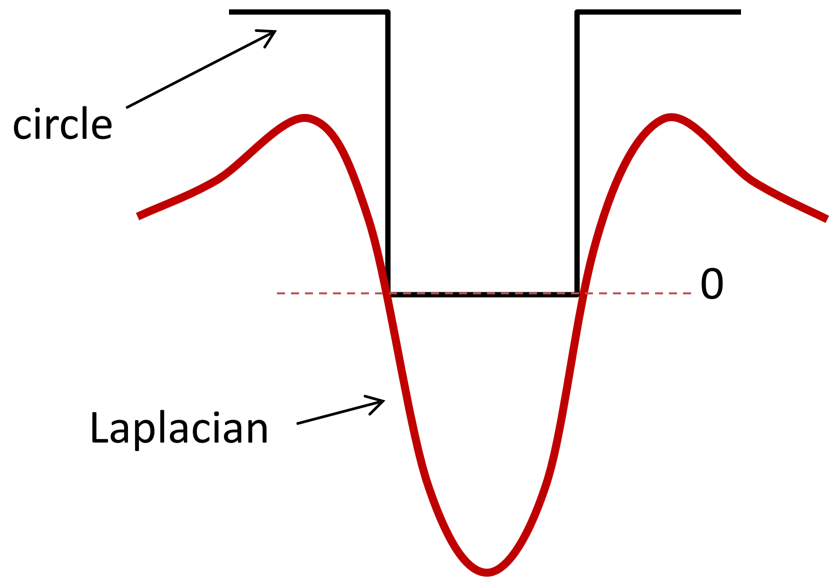
- At what scale does the Laplacian achieve a maximum response to a binary circle of radius r?
- To get maximum response, the zeros of the Laplacian have to be aligned with the circle
- The Laplacian is given by (up to scale):

$$L(x,y) = (x^2 + y^2 - 2\sigma^2)\, e^{-(x^2+y^2)/2\sigma^2}$$

- Therefore, the maximum response occurs at $\sigma = r/\sqrt{2}.$

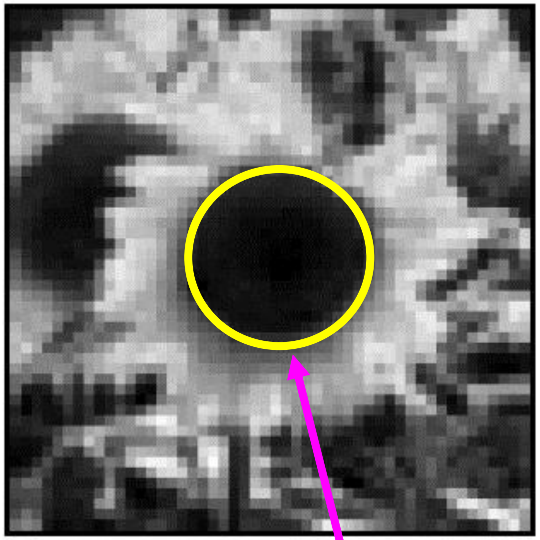


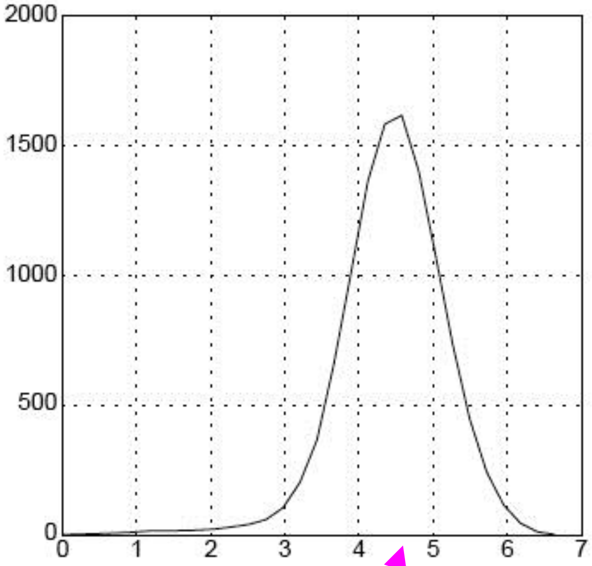
image



Characteristic scale

- Characteristic scale of a blob: the scale that produces peak of Laplacian response in the blob center





characteristic scale

T. Lindeberg (1998). "Feature detection with automatic scale selection." *International Journal of Computer Vision* **30** (2): pp 77--116.

Scale-space blob detector

1. Convolve image with scale-normalized Laplacian at several scales

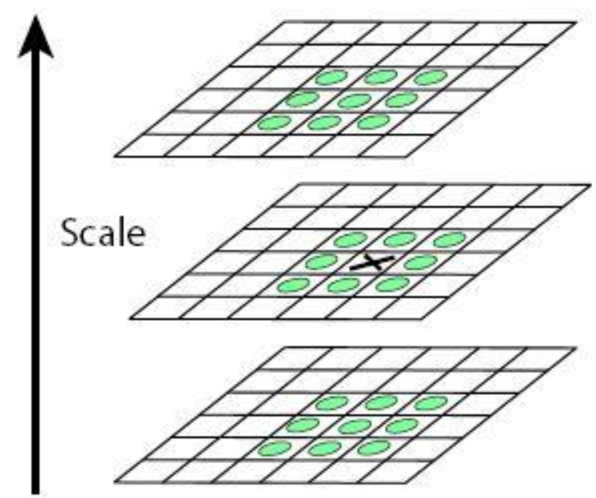
Scale-space blob detector: Example

Scale-space blob detector: Example

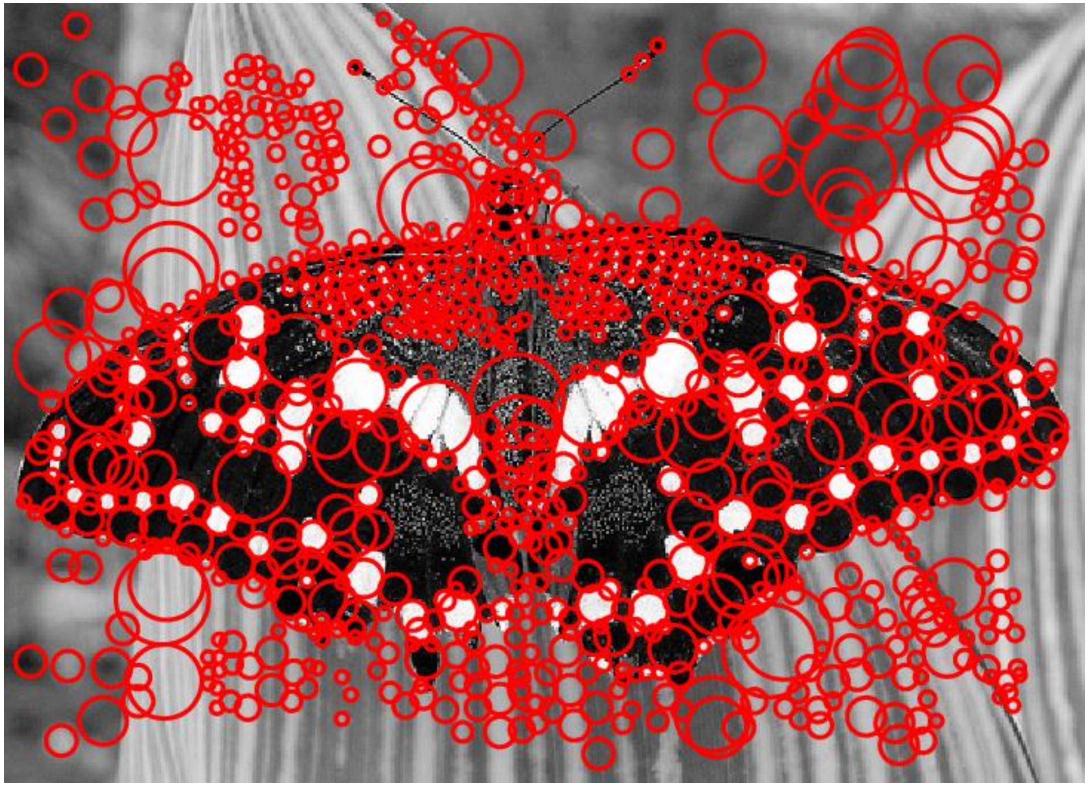
sigma = 9.5859

Scale-space blob detector

1. Convolve image with scale-normalized Laplacian at several scales
2. Find maxima of squared Laplacian response in scale-space



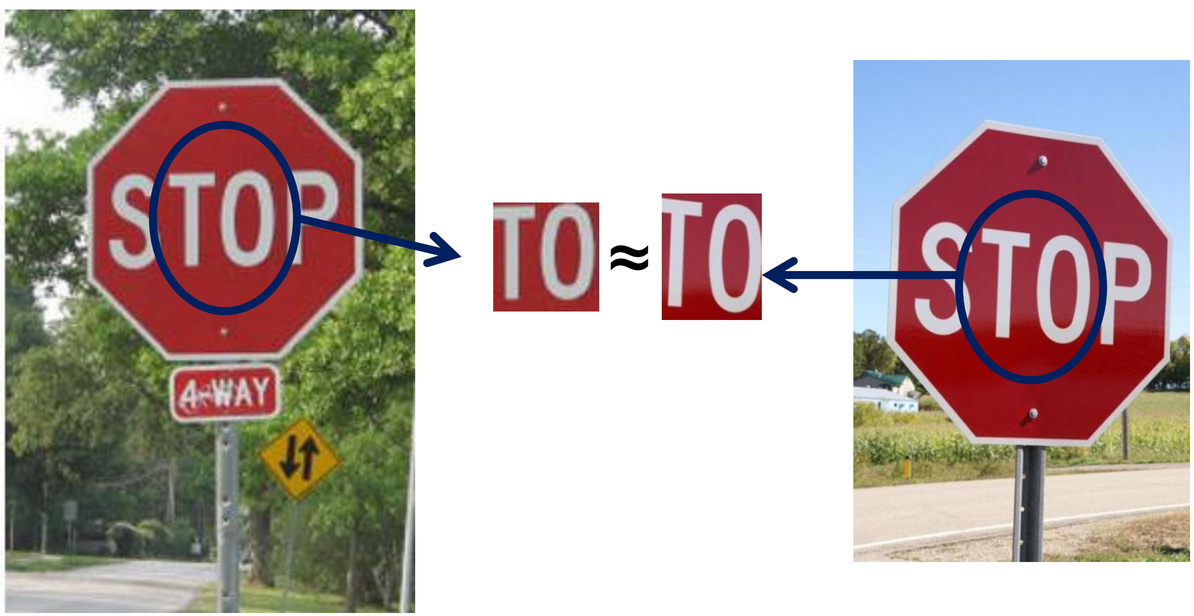
Scale-space blob detector: Example



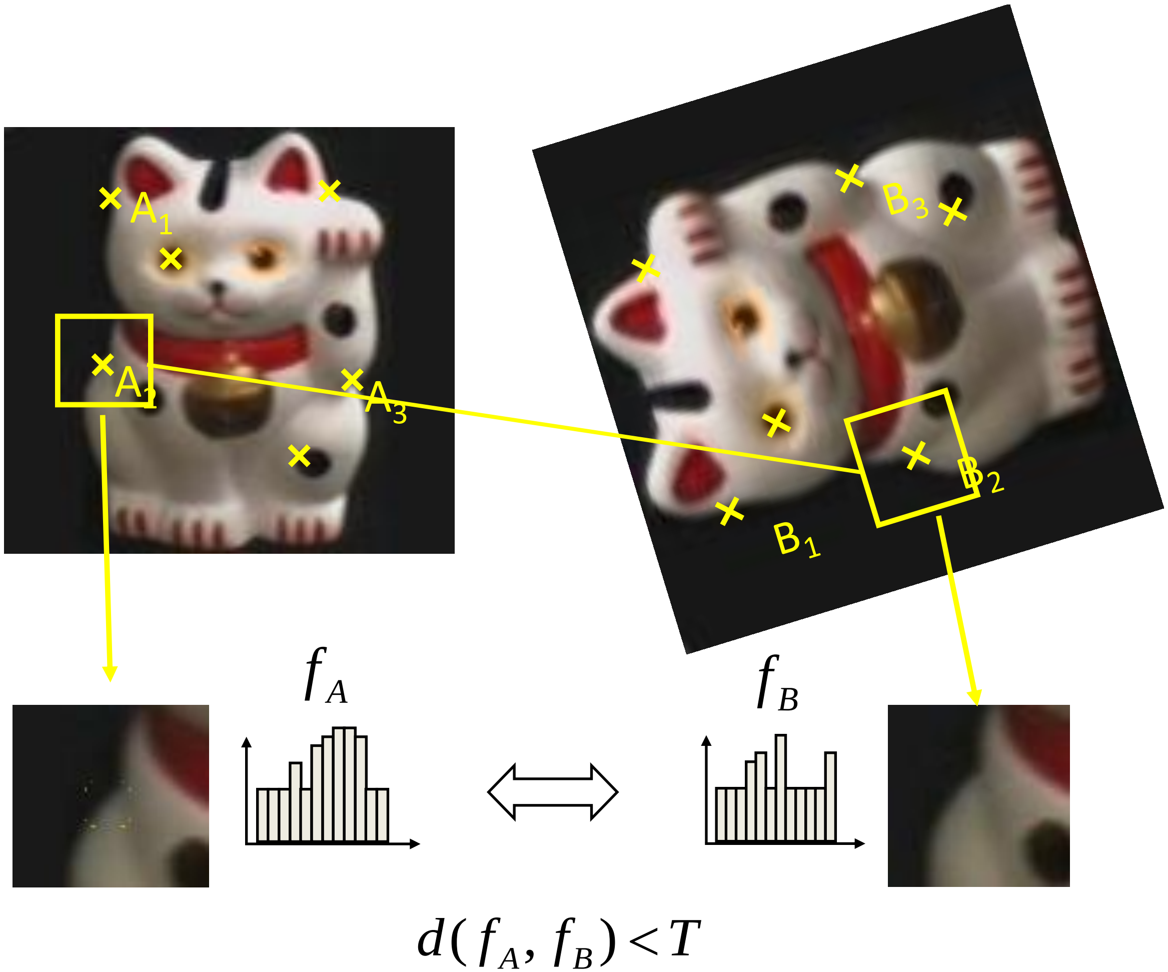
Feature Description/Matching

Correspondence and alignment

- Correspondence: matching points, patches, edges, or regions across images



Overview of Keypoint Matching



1. Find a set of distinctive key-points
2. Define a region around each keypoint
3. Extract and normalize the region content
4. Compute a local descriptor from the normalized region
5. Match local descriptors

K. Grauman, B. Leibe

Image representations

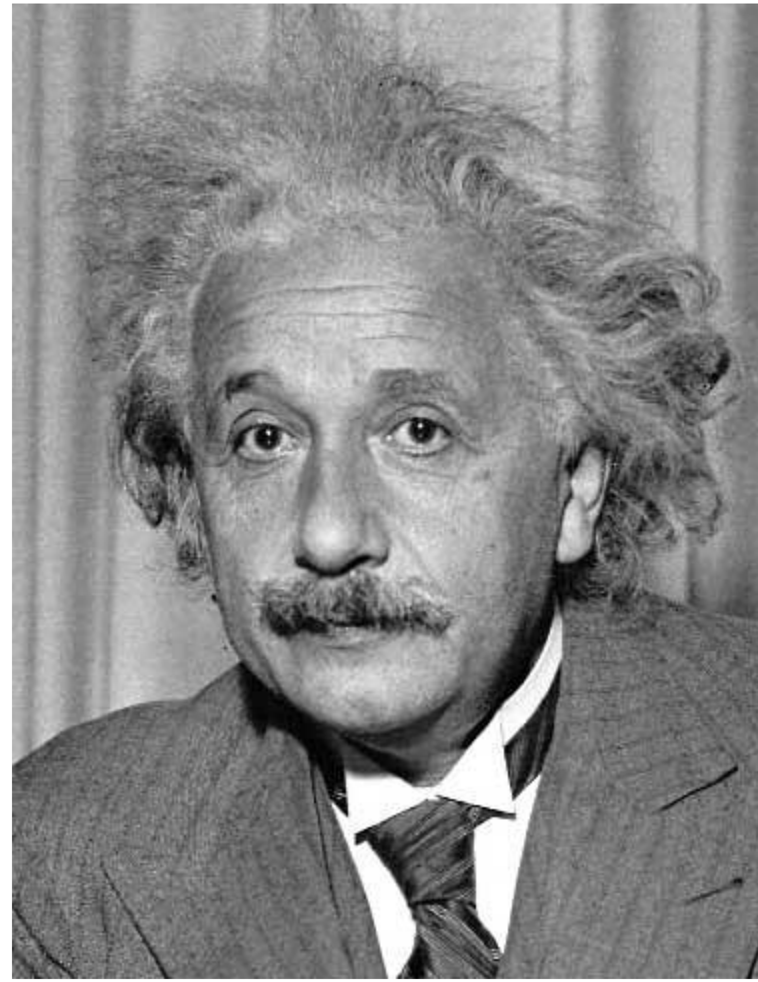
- Templates
  - Intensity, gradients, etc.

- Histograms
  - Color, texture, SIFT descriptors, etc.

Template matching

- Goal: find  in image

- Main challenge: What is a good similarity or distance measure between two patches?
  - Correlation
  - Zero-mean correlation
  - Sum Square Difference
  - Normalized Cross Correlation

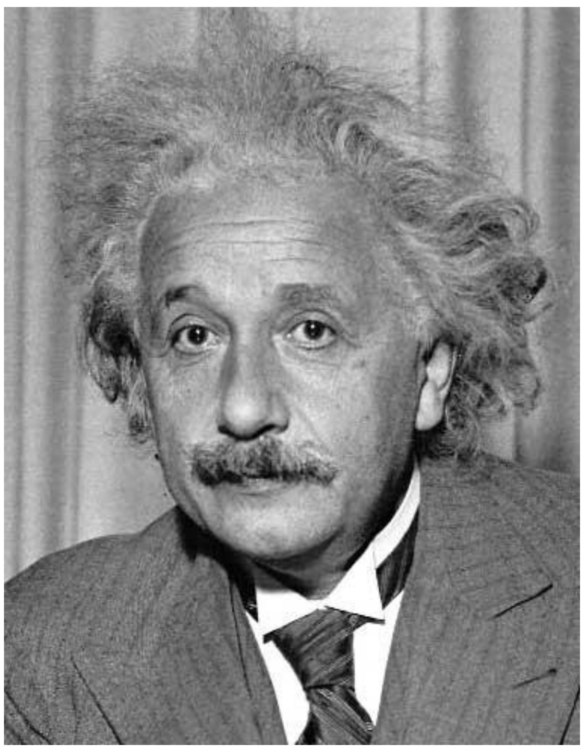


Matching with filters

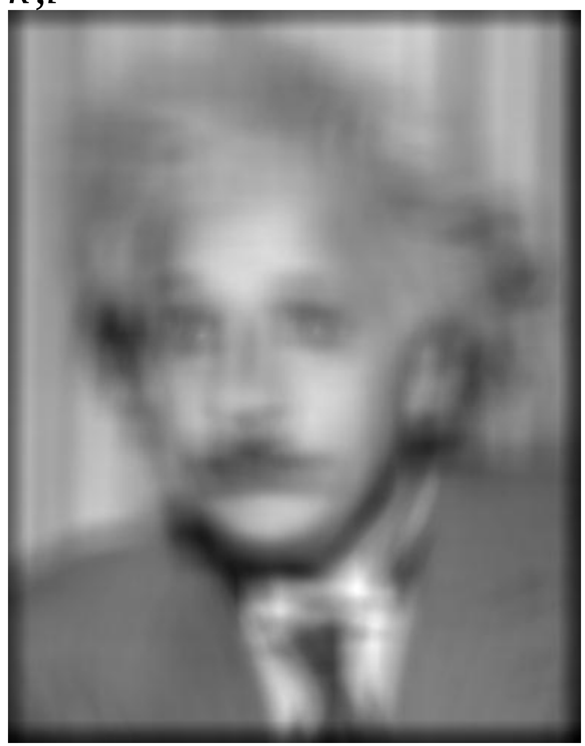
- Goal: find  in image
- Method 0: filter the image with eye patch

$$h[m,n]=\sum_{k,l} g[k,l]\; f[m+k,n+l]$$

f = image
g = filter



Input



Filtered Image

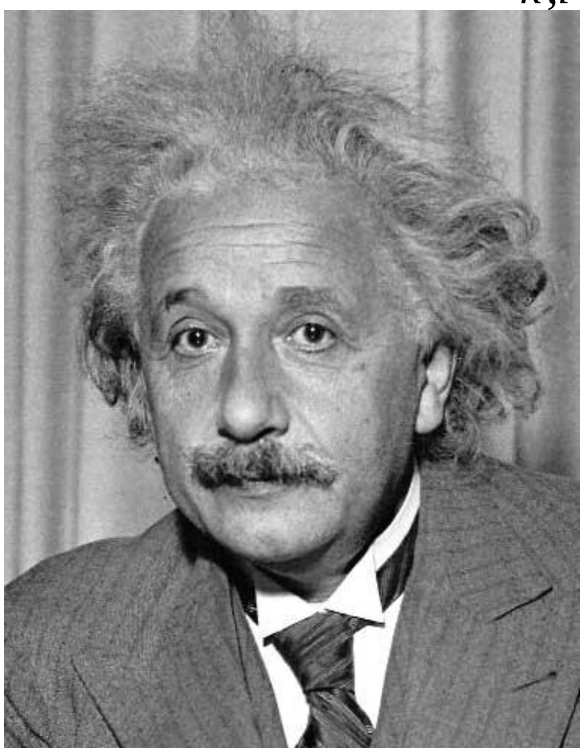
What went wrong?

Matching with filters

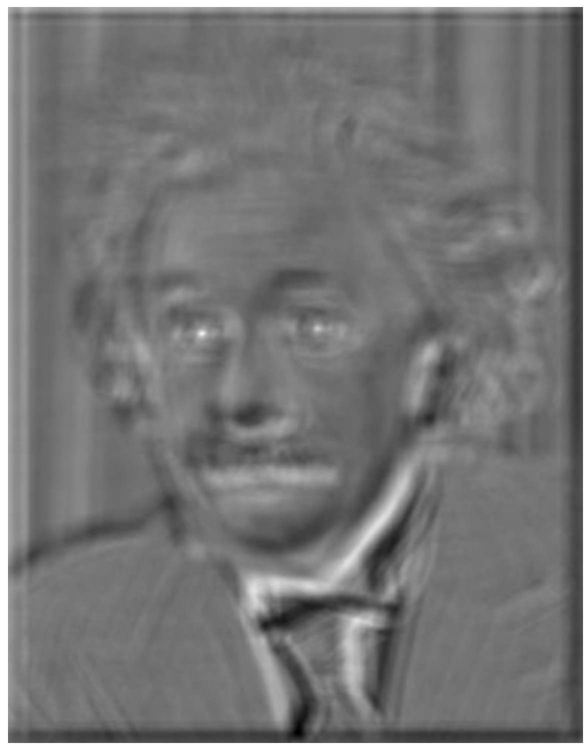
- Goal: find  in image
- Method 1: filter the image with zero-mean eye

$$h[m,n]=\sum_{k,l}(f[k,l]-\bar{f})(g[m+k,n+l])$$

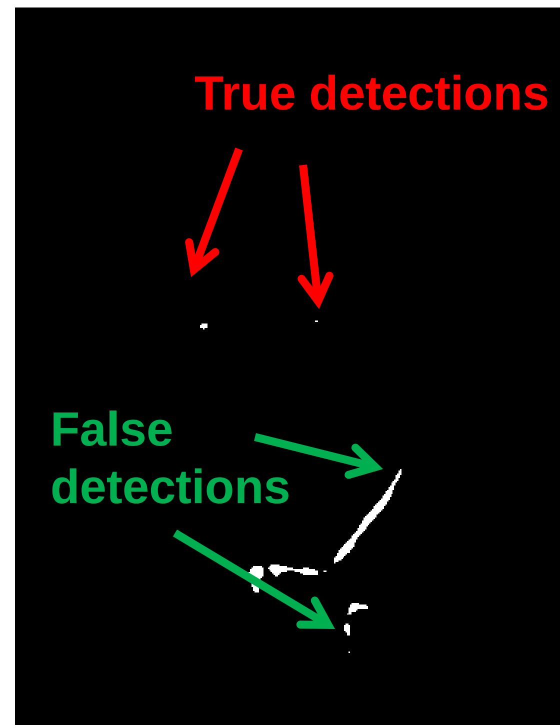
mean of f



Input



Filtered Image (scaled)

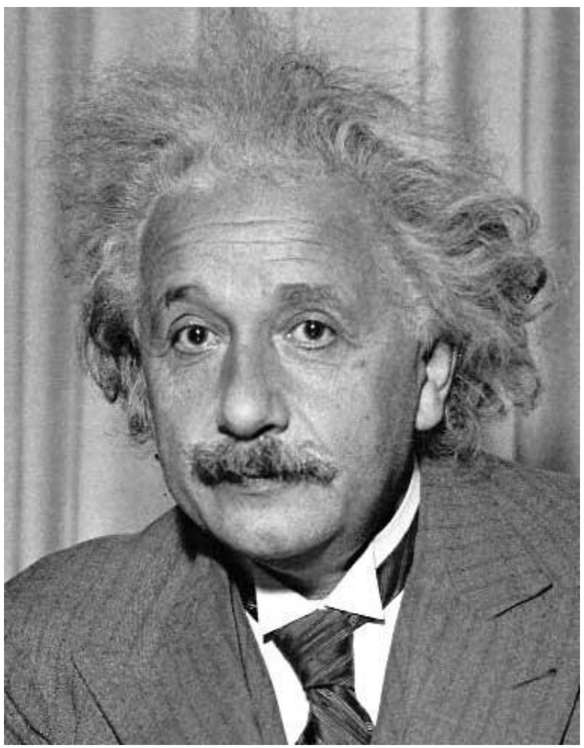


Thresholded Image

Matching with filters

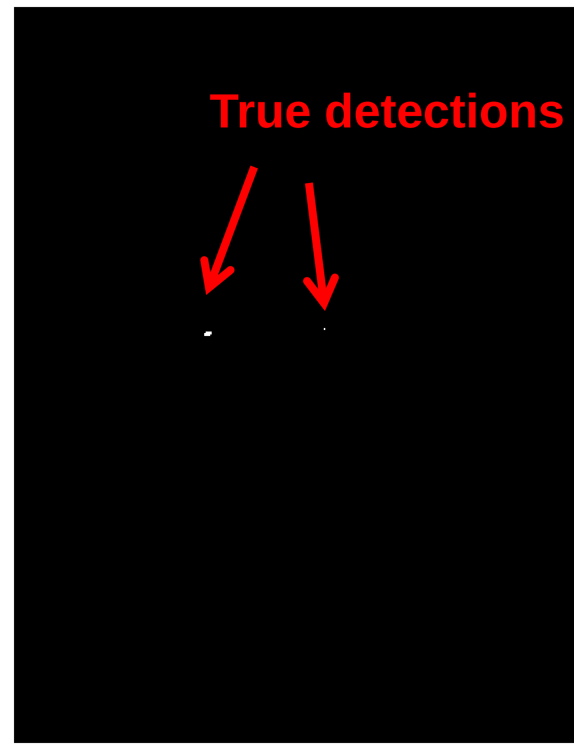
- Goal: find  in image
- Method 2: SSD

$$h[m,n] = \sum_{k,l} (g[k,l] - f[m+k, n+l])^2$$



Input

1- sqrt(SSD)



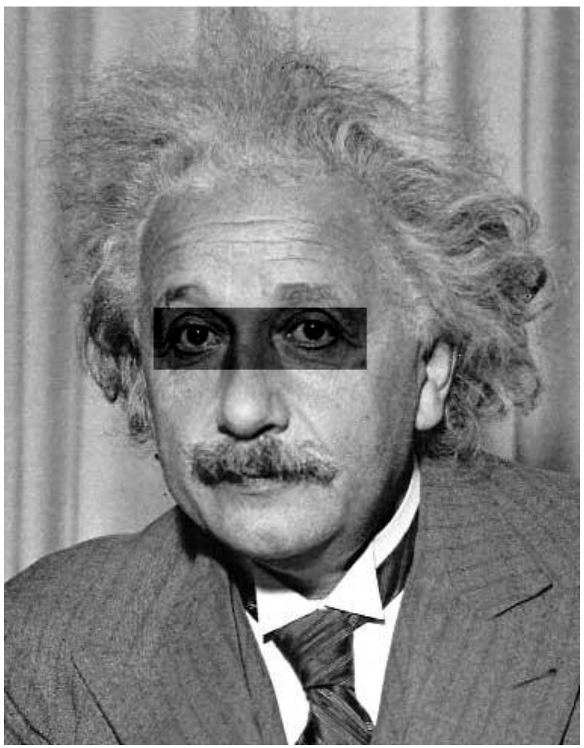
Thresholded Image

Matching with filters

What's the potential downside of SSD?

- Goal: find  in image
- Method 2: SSD

$$h[m,n]=\sum_{k,l}(g[k,l]-f[m+k,n+l])^2$$


Input


1- sqrt(SSD)

Matching with filters

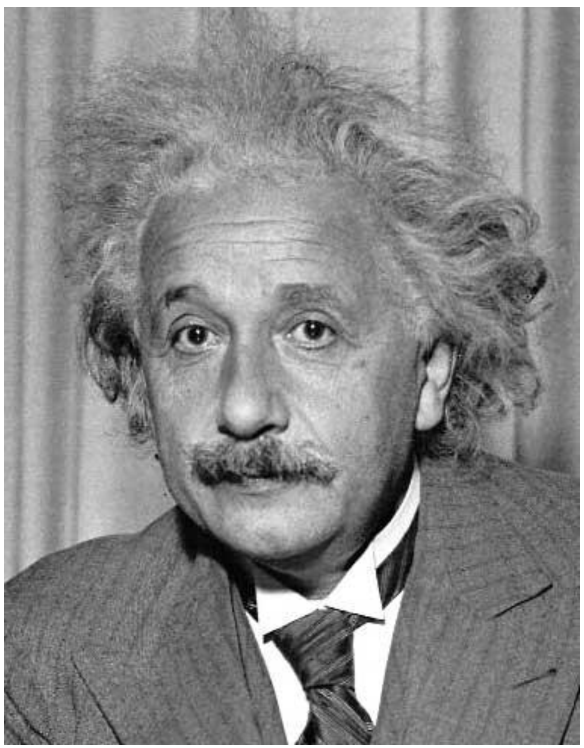
- Goal: find  in image
- Method 3: Normalized cross-correlation

$$h[m,n]=\frac{\sum_{k,l}(g[k,l]-\overset{\text{mean template}}{\overline{g}})(f[m-k,n-l]-\overset{\text{mean image patch}}{\bar{f}_{m,n}})}{\left(\sum_{k,l}(g[k,l]-\overline{g})^2\sum_{k,l}(f[m-k,n-l]-\bar{f}_{m,n})^2\right)^{0.5}}$$

Matlab: `normxcorr2(template, im)`

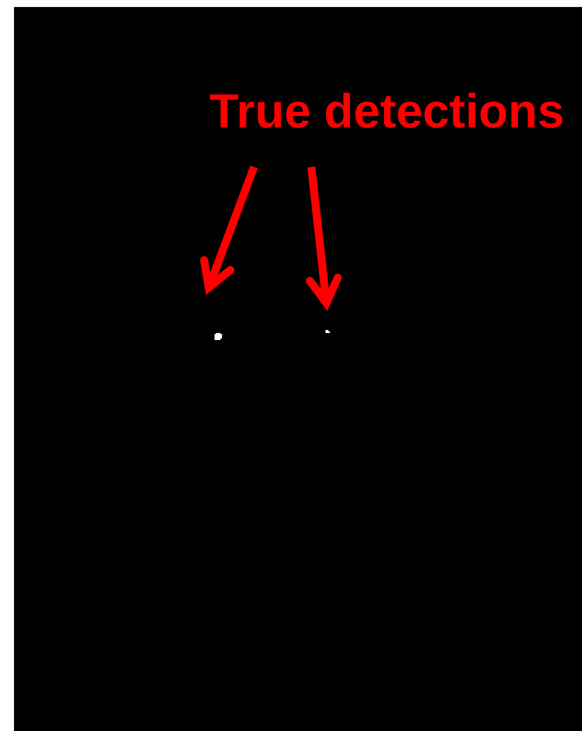
Matching with filters

- Goal: find  in image
- Method 3: Normalized cross-correlation



Input

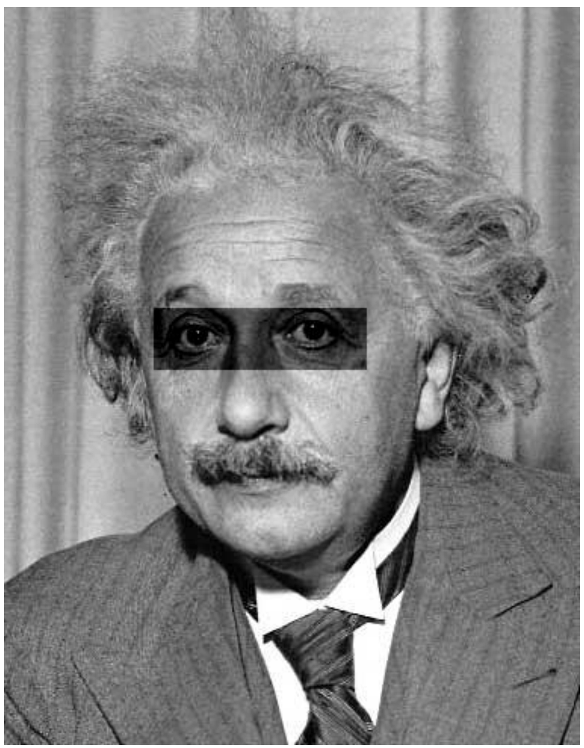
Normalized X-Correlation



Thresholded Image

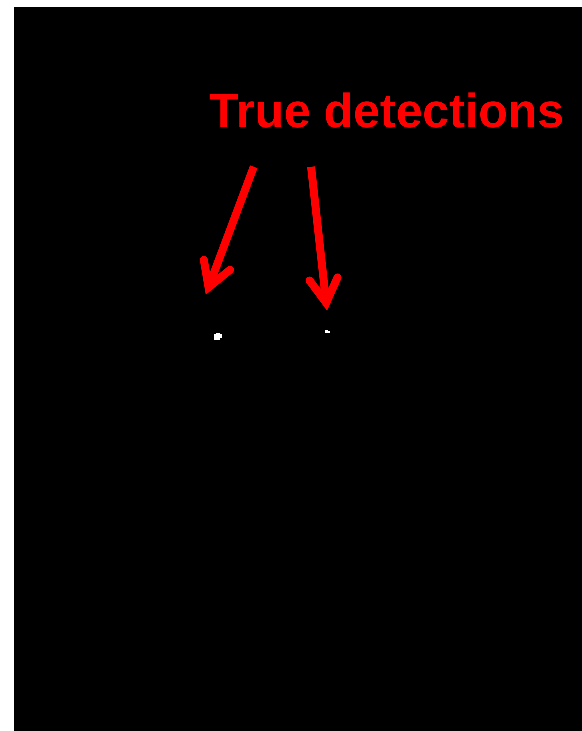
Matching with filters

- Goal: find  in image
- Method 3: Normalized cross-correlation



Input

Normalized X-Correlation



Thresholded Image

Q: What is the best method to use?

A: Depends

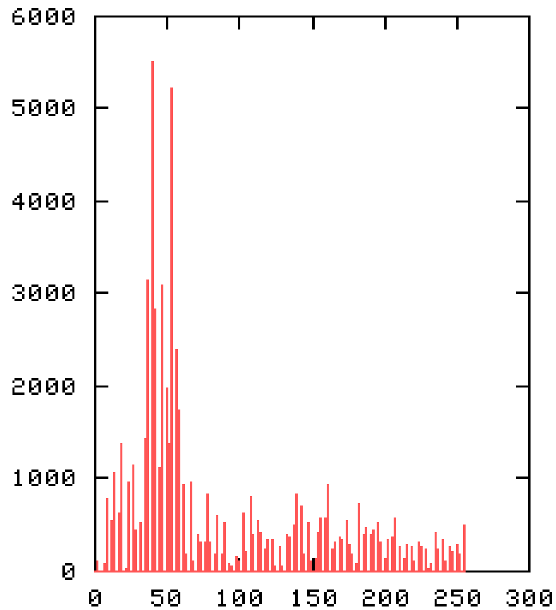
- SSD: faster, sensitive to overall intensity
- Normalized cross-correlation: slower, invariant to local average intensity and contrast
- But really, neither of these baselines are representative of modern recognition.

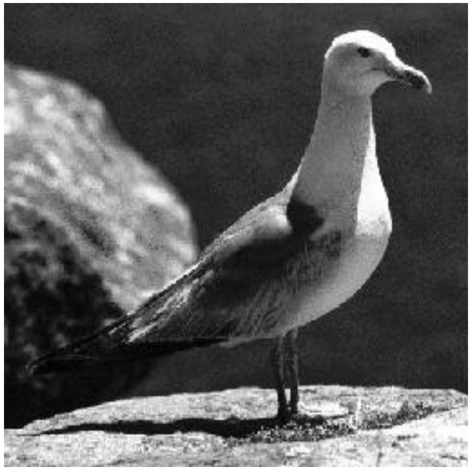
Image representations

- Templates
  - Intensity, gradients, etc.

- Histograms
  - Color, texture, SIFT descriptors, etc.

Image Representations: Histograms





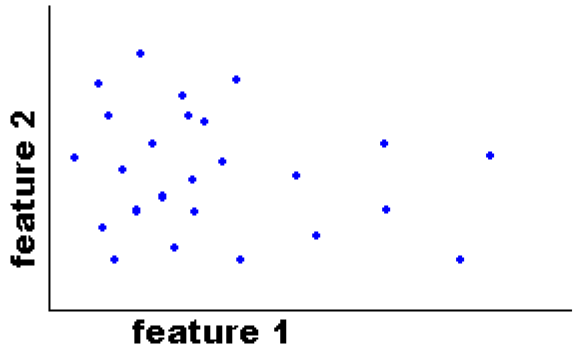
Global histogram

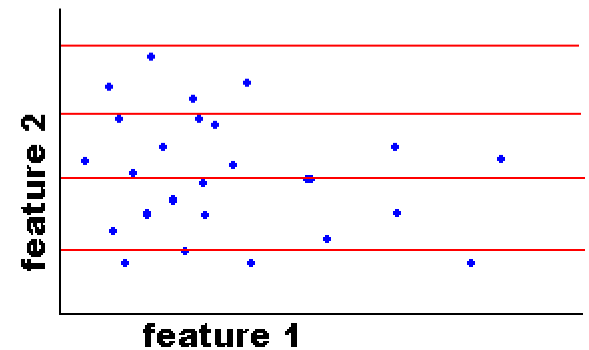
- Represent distribution of features
  - Color, texture, depth, …

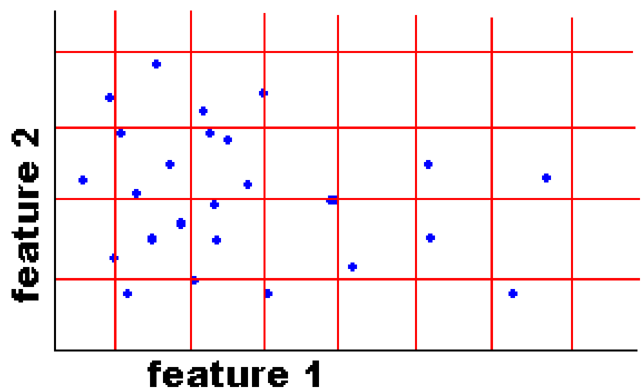
Images from Dave Kauchak

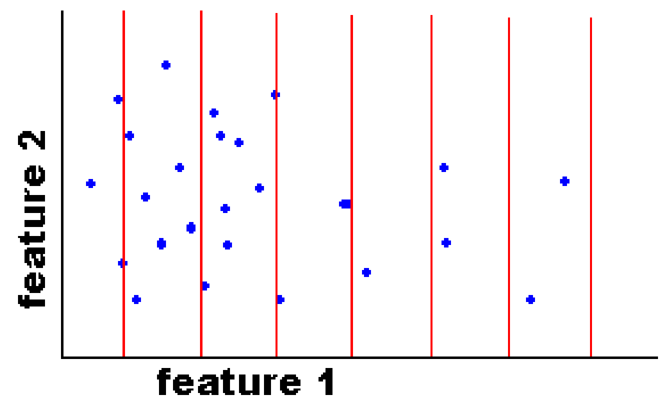
Image Representations: Histograms

Histogram: Probability or count of data in each bin









- Joint histogram
  - Requires lots of data
  - Loss of resolution to avoid empty bins

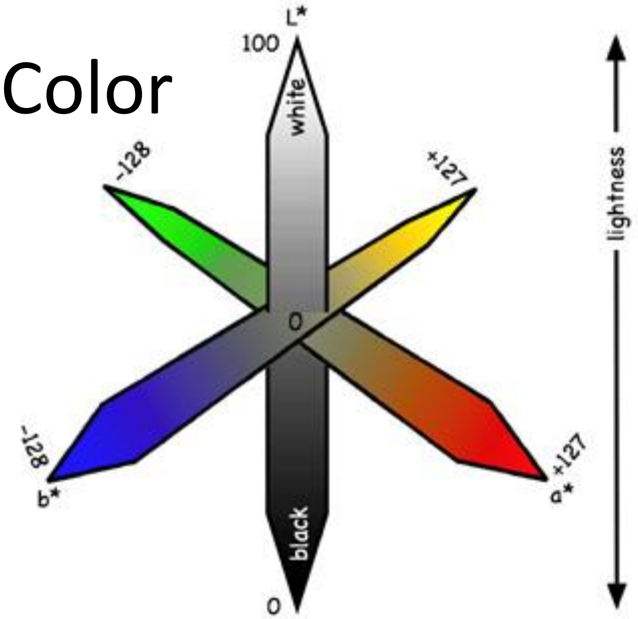
Marginal histogram

- Requires independent features
- More data/bin than joint histogram

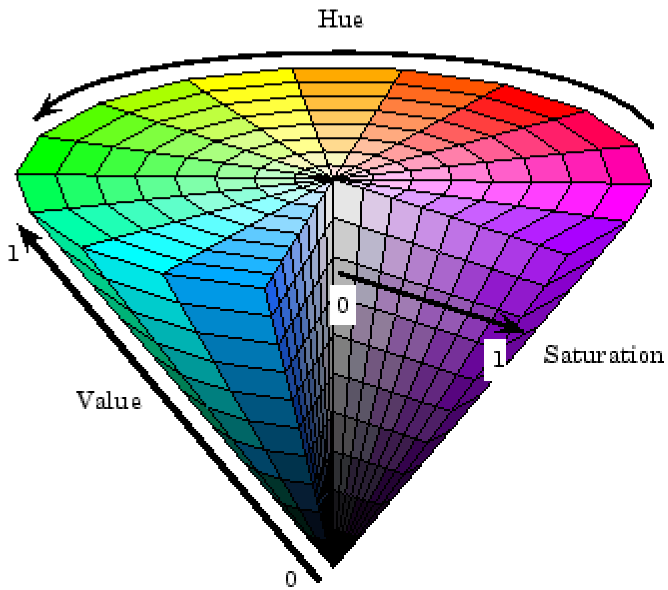
Images from Dave Kauchak

What kind of things do we compute histograms of?

- Color



L*a*b* color space

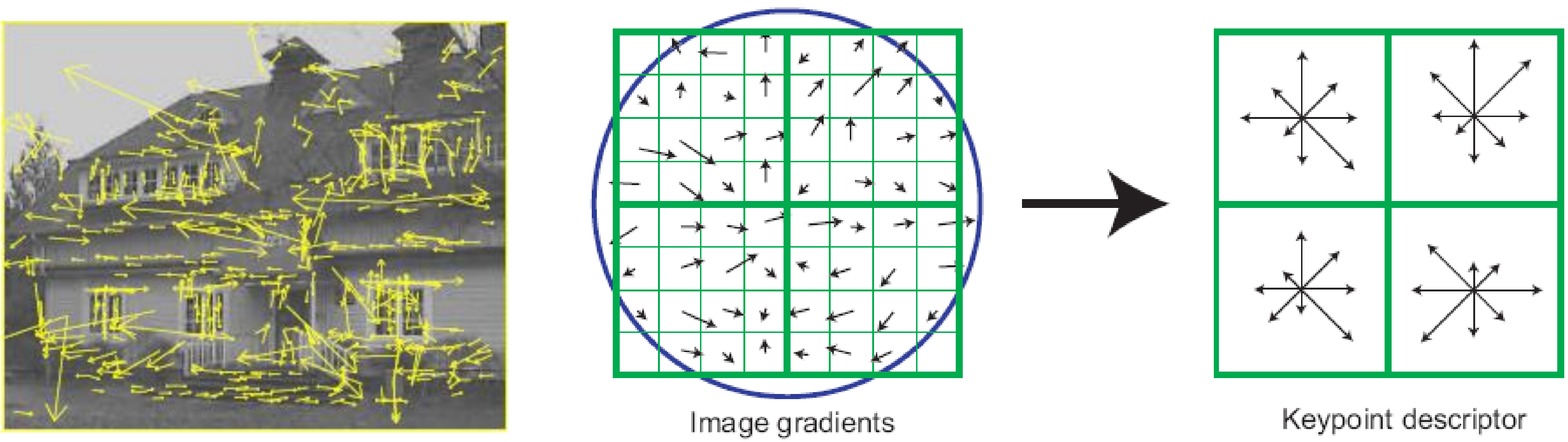


HSV color space

- Texture (filter banks or HOG over regions)

SIFT

- Histograms of oriented gradients



SIFT – Lowe IJCV 2004

SIFT vector formation

- Computed on rotated and scaled version of window according to computed orientation & scale

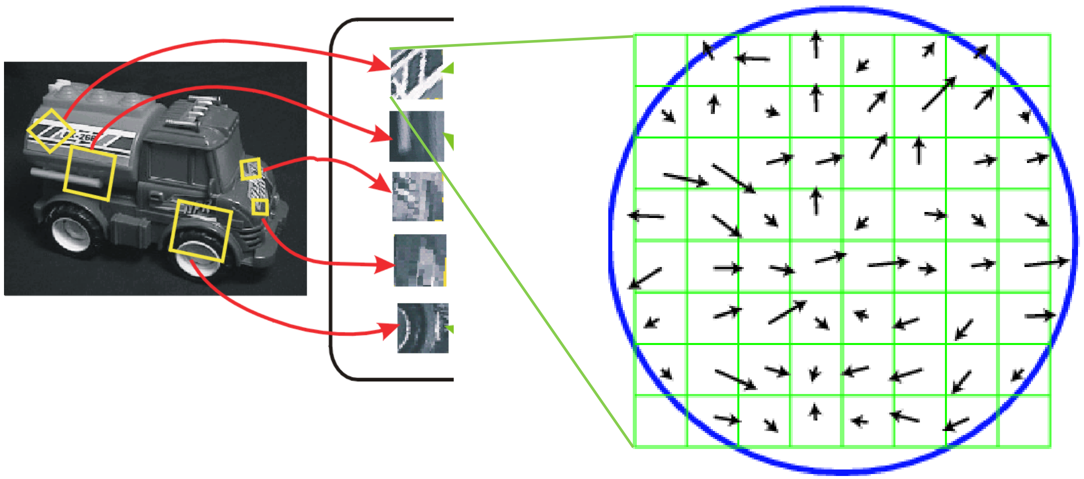
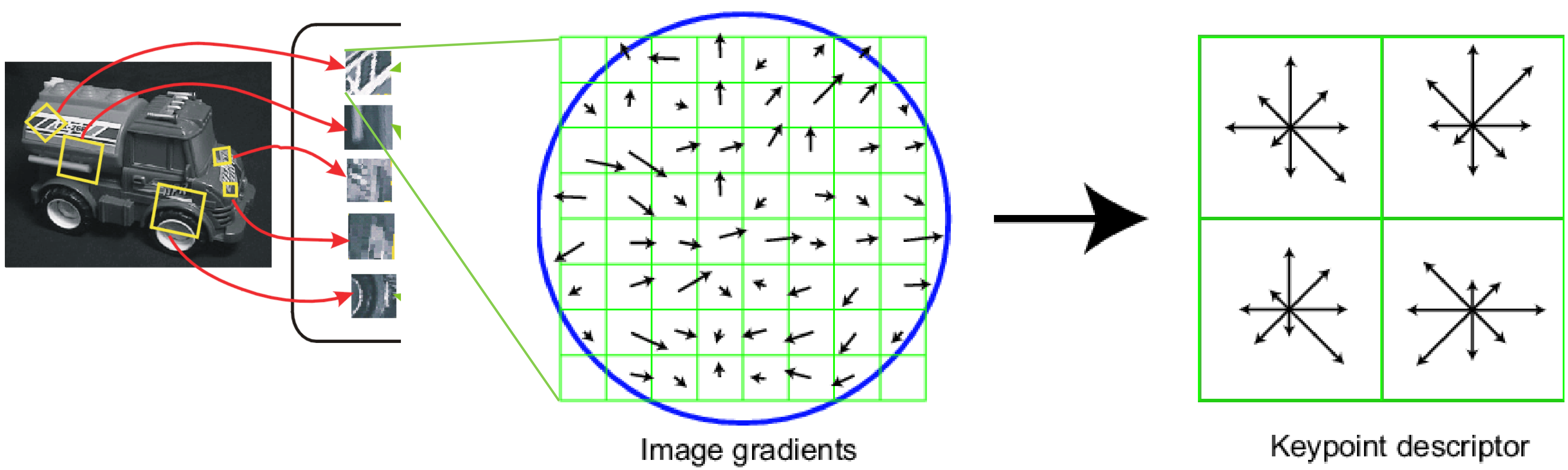


Image gradients

SIFT vector formation

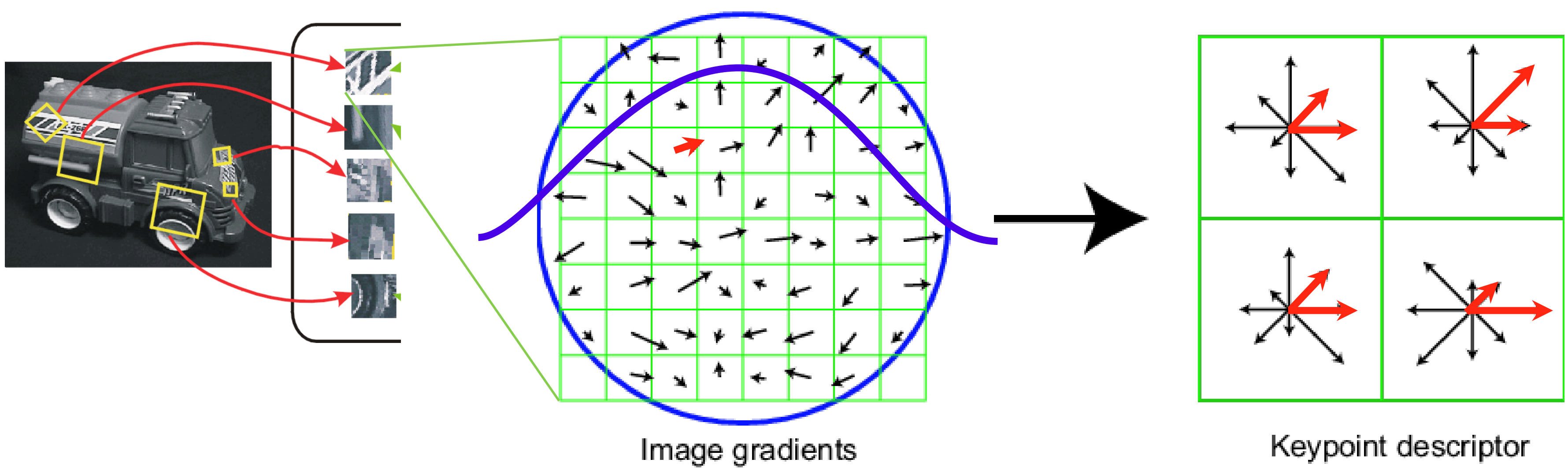
- 4x4 array of gradient orientation histogram weighted by magnitude
- 8 orientations x 4x4 array = 128 dimensions
- Motivation: some sensitivity to spatial layout, but not too much.



showing only 2x2 here but is 4x4

Ensure smoothness

- Gaussian weight
- Trilinear interpolation
  - a given gradient contributes to 8 bins: 4 in space times 2 in orientation



Reduce effect of illumination

- 128-dim vector normalized to 1
- Threshold gradient magnitudes to avoid excessive influence of high gradients
  - after normalization, clamp gradients >0.2
  - renormalize