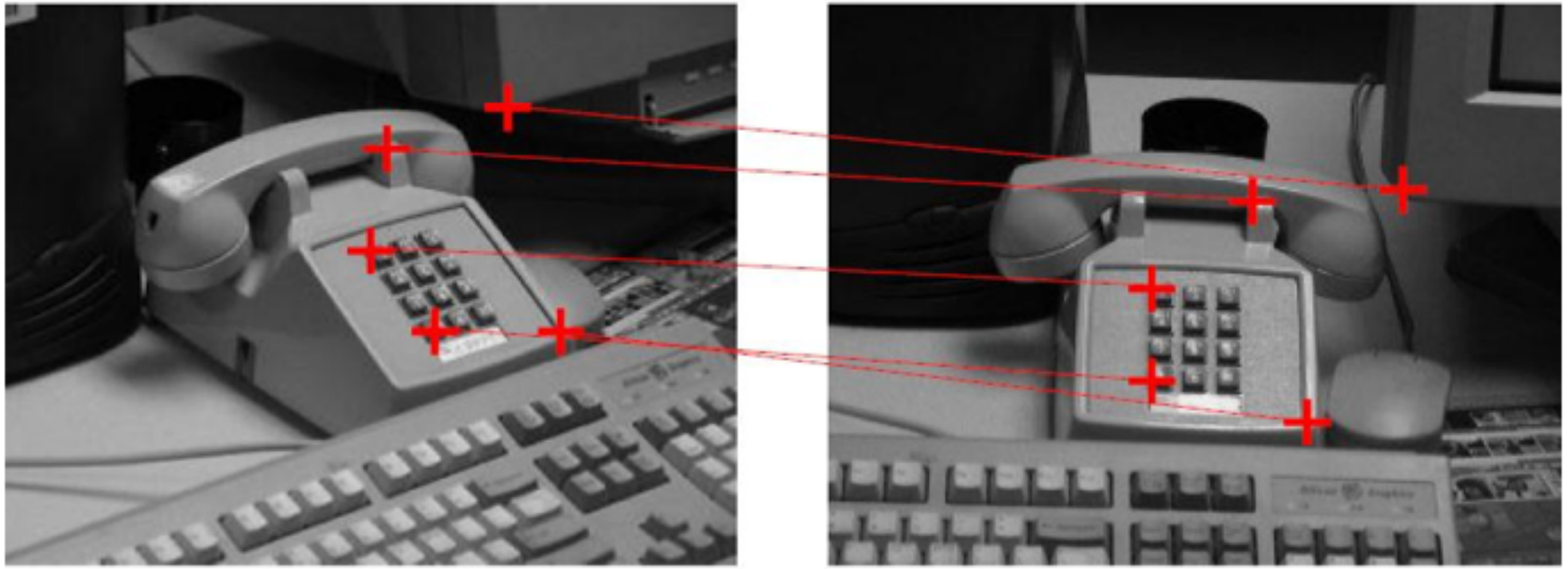
Detectors: Harris, Scale invariance, Harris-Laplace, affine Harris-Laplace

Gary Overett (Slides adapted from CMU 16-720 2014)

Example: Finding Correspondences Between Images



- First step toward 3-D reconstruction: Find correspondences between feature points in two images of a scene

- Object recognition: Find correspondences between feature points in “training” and “test” image

Rome in a Day!

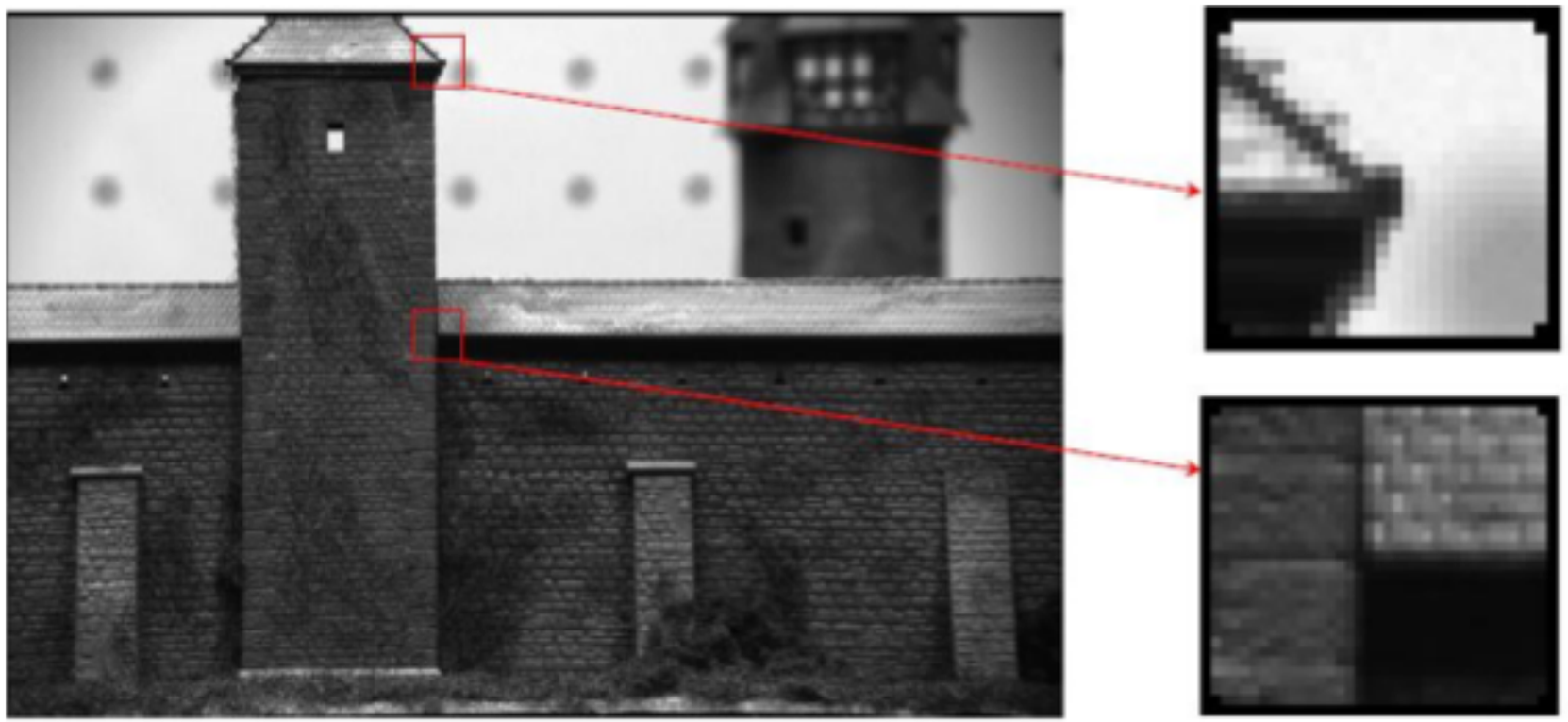
- 2,106 images, 819,242 points
- http://grail.cs.washington.edu/projects/rome/
- Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz and Richard Szeliski International Conference on Computer Vision, 2009, Kyoto, Japan.

https://youtu.be/qYaU1GeEiR8?list=PLDFDB5B8C80DB3AD6

What makes a good interest point?

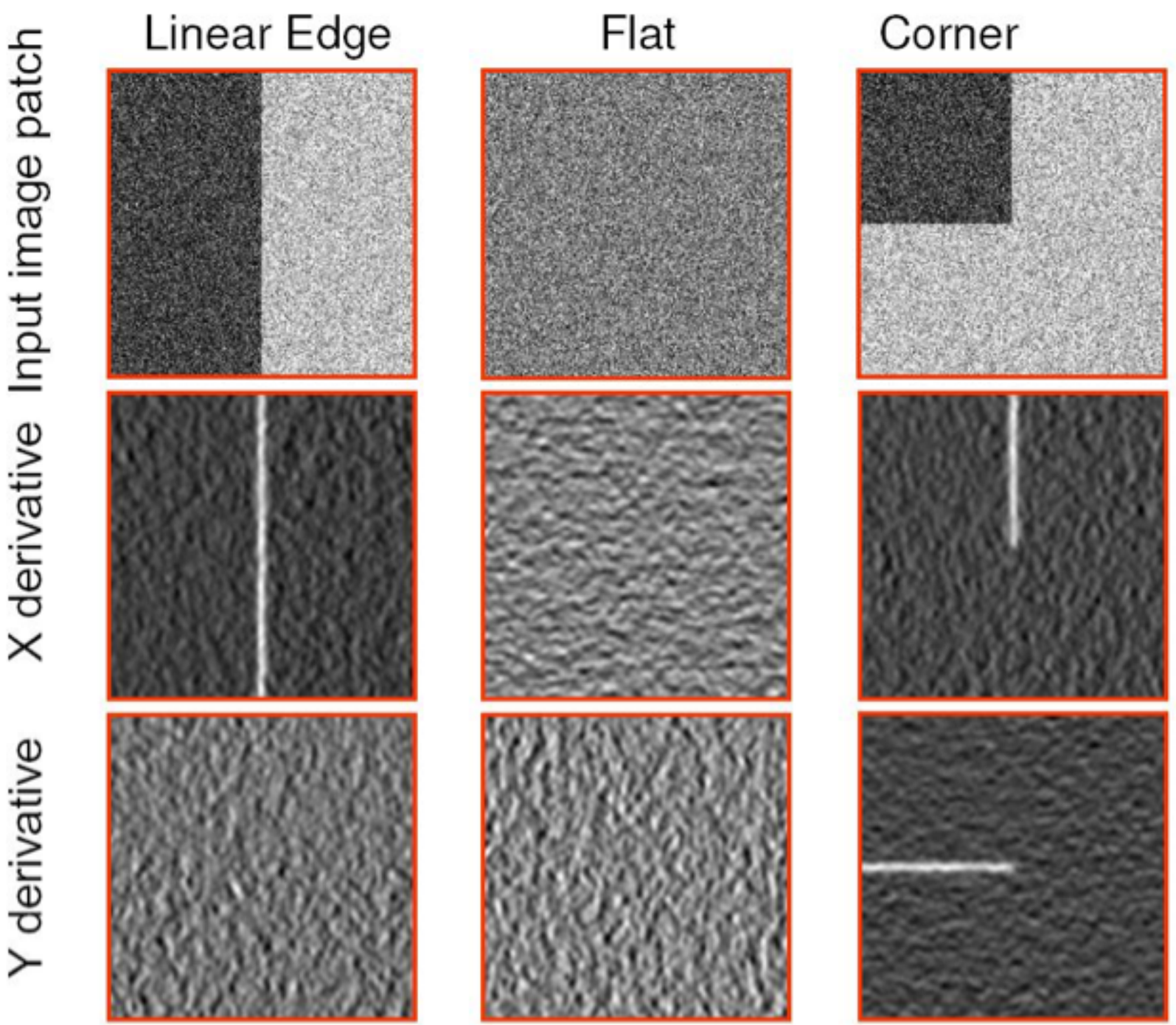
- Stable across viewpoints, illumination changes, etc
- Computationally efficient (we need to find many)
- Local Uniqueness

Interest Points - Corners!

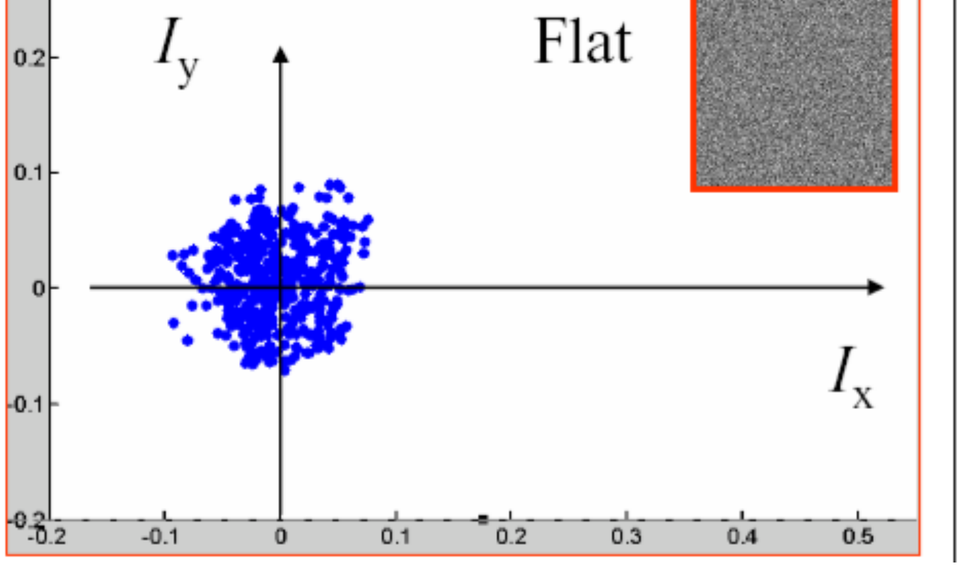


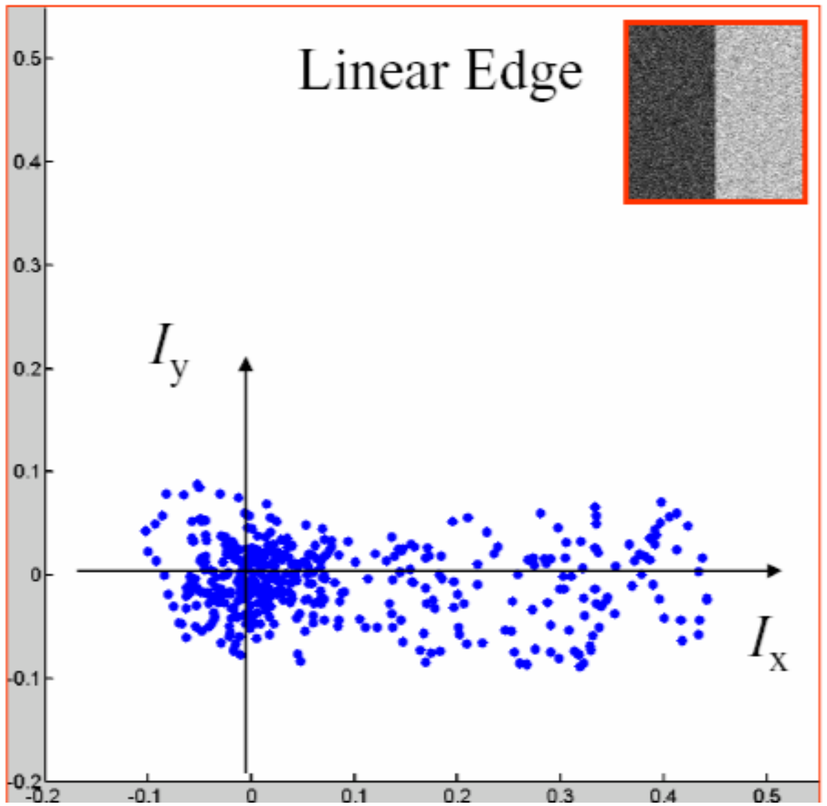
- Intuitively, junctions of contours.
- Generally more stable features over changes of viewpoint
- Intuitively, large variations in the neighborhood of the point in all directions

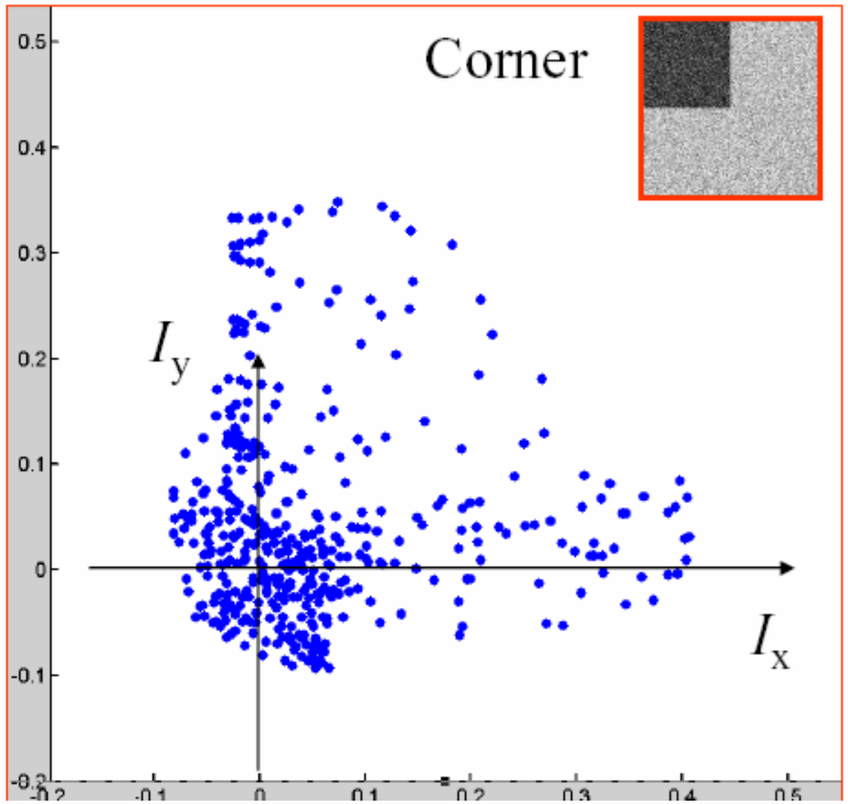
Behold! The Harris Corner Detector…


Linear Edge
Flat
Corner
Input image patch
X derivative
Y derivative

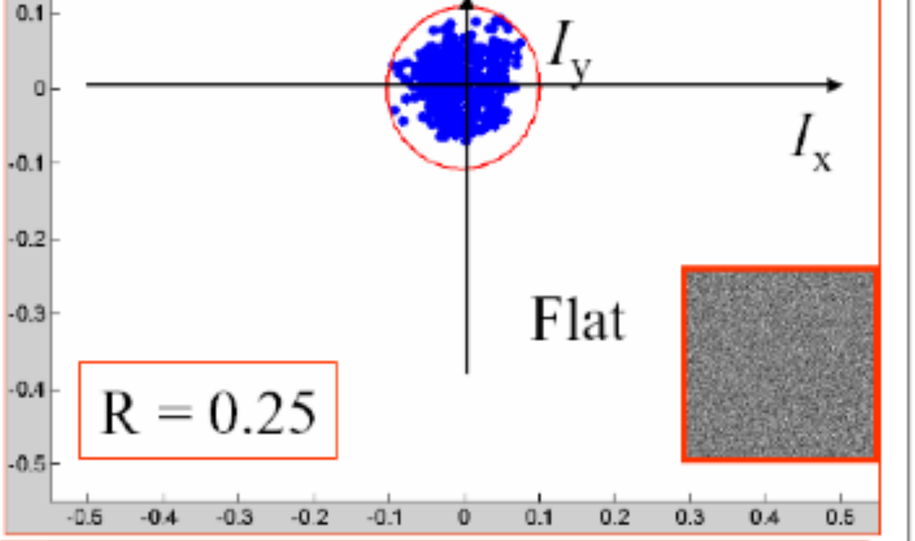
The distribution of the x and y derivatives is very different for all three types of patches

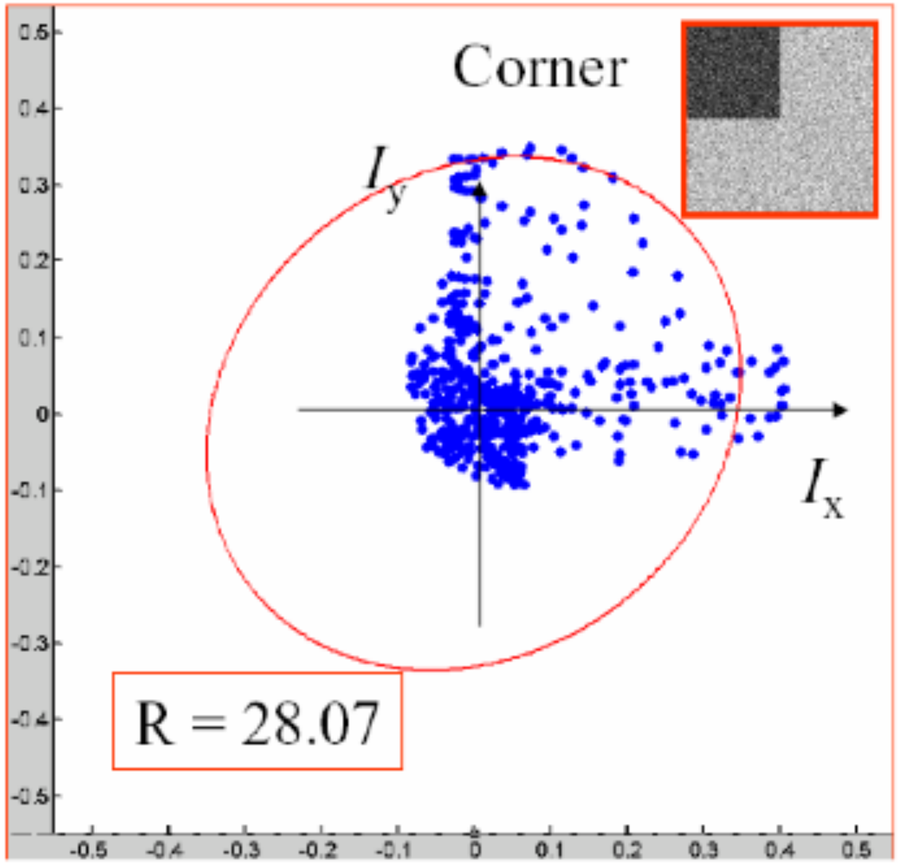


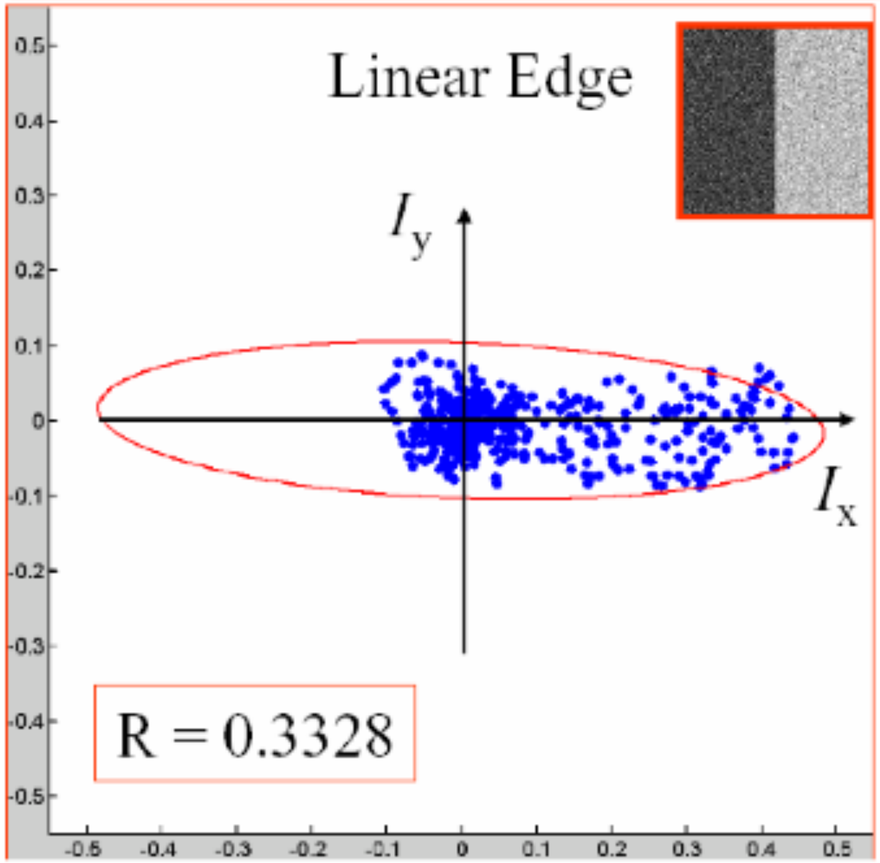




The distribution of x and y derivatives can be characterized by the shape and size of the principal component ellipse







Nice idea, how do we capture it mathematically?

- The distribution of gradients in a neighborhood W is represented by the inertia (shape, Harris) matrix:

$$\boldsymbol{H} = \begin{bmatrix} \sum_{W} \boldsymbol{I}_x^2 & \sum_{W} \boldsymbol{I}_x \boldsymbol{I}_y \\ \sum_{W} \boldsymbol{I}_x \boldsymbol{I}_y & \sum_{W} \boldsymbol{I}_y^2 \end{bmatrix}$$

- Elongations of the distribution = ????????

Nice idea, how do we capture it mathematically?

- The distribution of gradients in a neighborhood *W* is represented by the inertia (shape, Harris) matrix:

$$\boldsymbol{H} = \begin{bmatrix} \sum_{W} \boldsymbol{I}_x^2 & \sum_{W} \boldsymbol{I}_x \boldsymbol{I}_y \\ \sum_{W} \boldsymbol{I}_x \boldsymbol{I}_y & \sum_{W} \boldsymbol{I}_y^2 \end{bmatrix}$$

- Elongations of the distribution = Eigenvalues of H (of course!)

Nice idea, how do we capture it mathematically?

- The distribution of gradients in a neighborhood *W* is represented by the inertia (shape, Harris) matrix:

$$\boldsymbol{H} = \begin{bmatrix} \sum_{W} \boldsymbol{I}_x^2 & \sum_{W} \boldsymbol{I}_x \boldsymbol{I}_y \\ \sum_{W} \boldsymbol{I}_x \boldsymbol{I}_y & \sum_{W} \boldsymbol{I}_y^2 \end{bmatrix}$$

- Elongations of the distribution = Eigenvalues of H (of course!)
  - We want $\lambda_{\min}, \lambda_{\max}$ to be approx. equal AND large

Harris Detector and variants

- we want $\lambda_{\min}, \lambda_{\max}$ to be equal and large

$$R = 4\frac{\lambda_{\min}\lambda_{\max}}{(\lambda_{\min} + \lambda_{\max})^2}$$

- R=1 if $\lambda_{\min}, \lambda_{\max}$ are equal but keep only ones with large $\lambda_{\max}$

Harris Detector and variants

- we want $\lambda_{\min}, \lambda_{\max}$ to be equal and large

$$R = 4\frac{\lambda_{\min}\lambda_{\max}}{(\lambda_{\min} + \lambda_{\max})^2}$$

- R=1 if $\lambda_{\min}, \lambda_{\max}$ are equal but keep only ones with large $\lambda_{\max}$

$$R = \frac{\lambda_{\min}\lambda_{\max}}{(\lambda_{\min} + \lambda_{\max})}$$

$$R = \lambda_{\min}\lambda_{\max} - k(\lambda_{\min} + \lambda_{\max})$$

Computationally Efficient Definition

- For any symmetric matrix H:
  - Det(H) = $\lambda_{\min} \lambda_{\max}$
  - Trace(H) = $\lambda_{\min} + \lambda_{\max}$ (The trace is the sum of the diagonal of H)

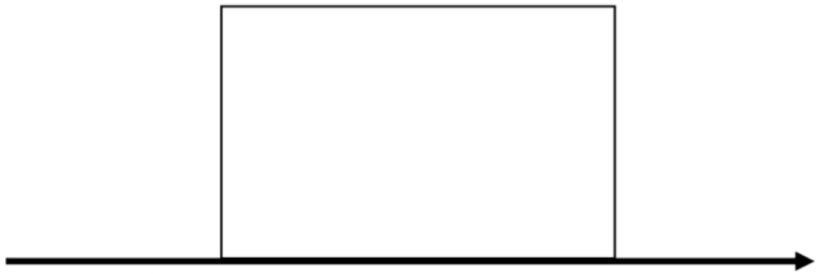
$$R = 4\frac{Det(H)}{Trace(H)^2}$$

$$R = \frac{Det(H)}{Trace(H)}$$

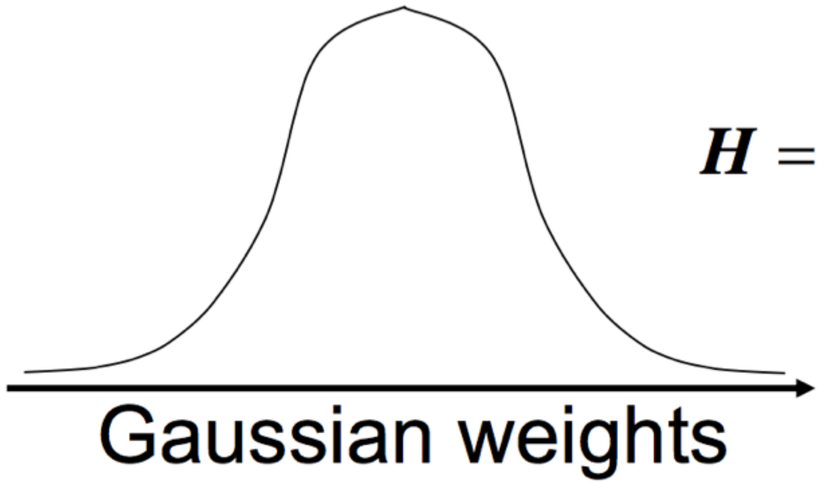
$$R = Det(H) - kTrace(H)^2$$

Computation of H and gradients

- I_x means convolution with Gaussian of σ
- H should computed with different weights → Convolution with Gaussian



$$\boldsymbol{H} = \begin{bmatrix} \sum_W \boldsymbol{I}_x^2 & \sum_W \boldsymbol{I}_x \boldsymbol{I}_y \\ \sum_W \boldsymbol{I}_x \boldsymbol{I}_y & \sum_W \boldsymbol{I}_y^2 \end{bmatrix}$$

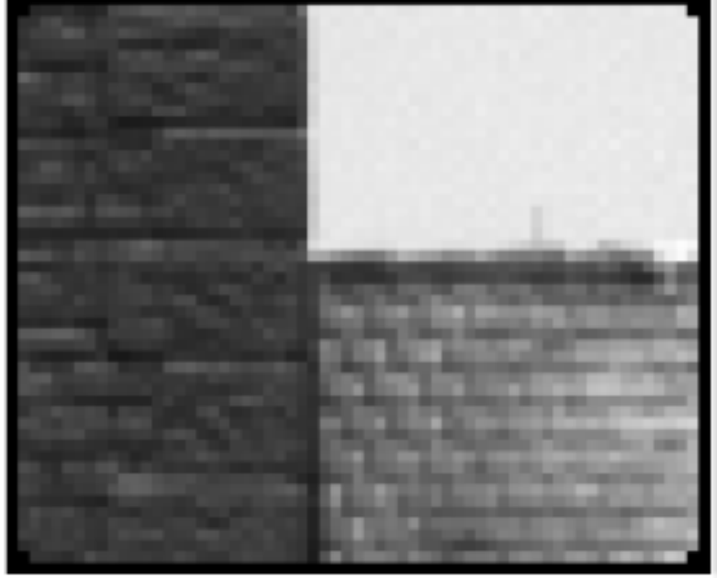


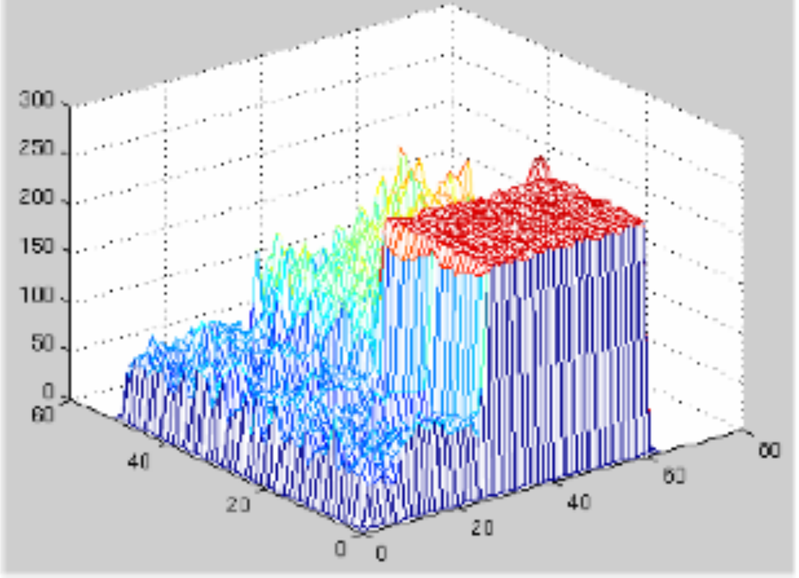
$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{G}_\sigma * \boldsymbol{I}_x^2 & \boldsymbol{G}_\sigma * \boldsymbol{I}_x \boldsymbol{I}_y \\ \boldsymbol{G}_\sigma * \boldsymbol{I}_x \boldsymbol{I}_y & \boldsymbol{G}_\sigma * \boldsymbol{I}_y^2 \end{bmatrix} = \boldsymbol{G}_\sigma * \begin{bmatrix} \boldsymbol{I}_x^2 & \boldsymbol{I}_x \boldsymbol{I}_y \\ \boldsymbol{I}_x \boldsymbol{I}_y & \boldsymbol{I}_y^2 \end{bmatrix}$$

Input

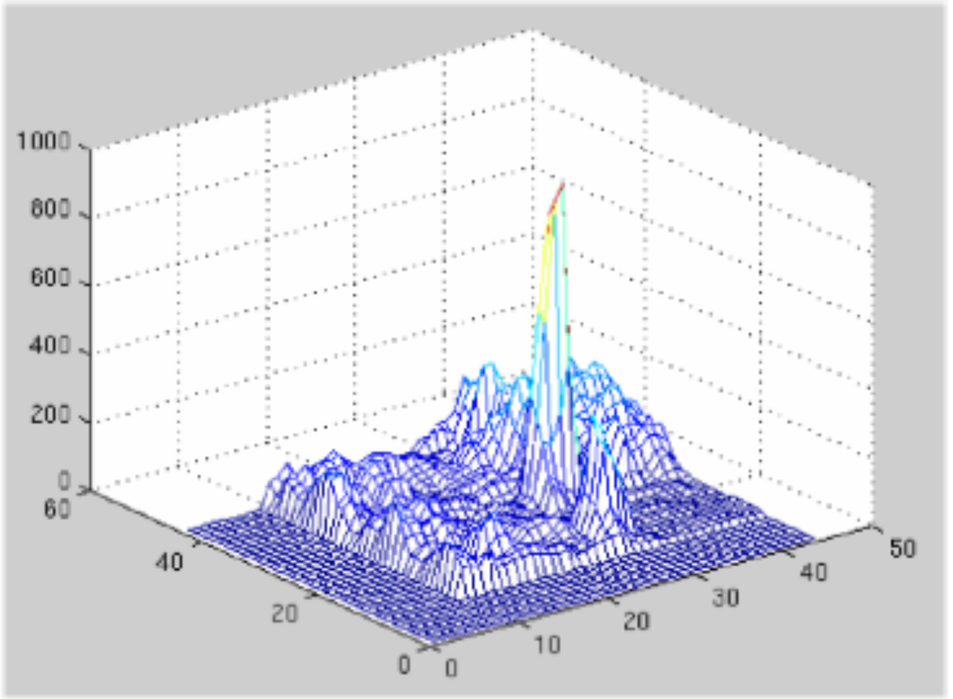
Response of interest point operator

Input





Response of interest point operator



Interest point detection example

Interest points example

1. Compute x and y derivatives of image

$$I_x = G_\sigma^x * I \quad I_y = G_\sigma^y * I$$

2. Compute products of derivatives at every pixel

$$I_{x2} = I_x.I_x \quad I_{y2} = I_y.I_y \quad I_{xy} = I_x.I_y$$

3. Compute the sums of the products of derivatives at each pixel

$$S_{x2} = G_{\sigma\prime} * I_{x2} \quad S_{y2} = G_{\sigma\prime} * I_{y2} \quad S_{xy} = G_{\sigma\prime} * I_{xy}$$

4. Define at each pixel (x, y) the matrix

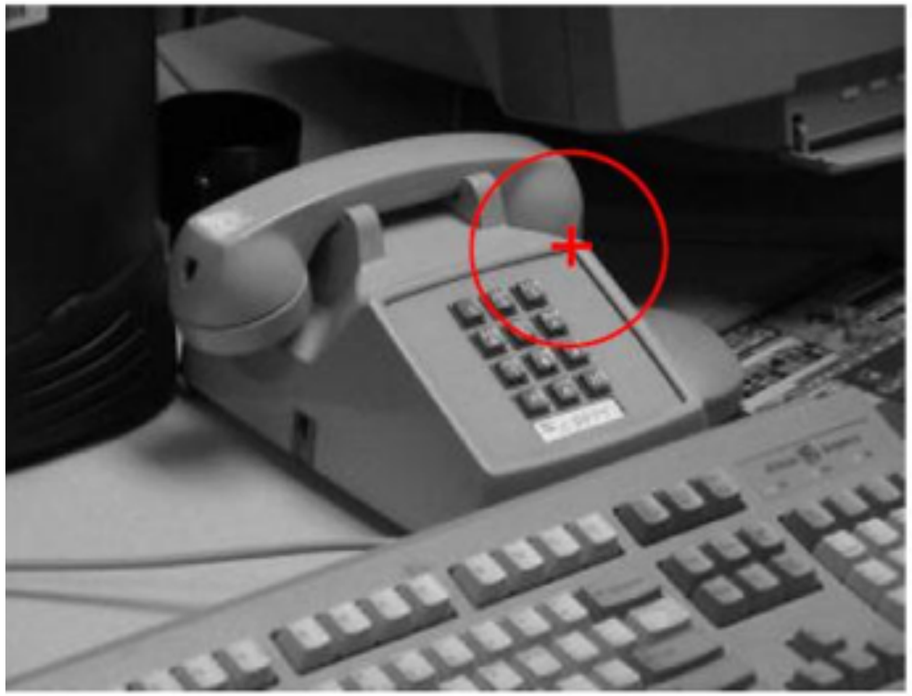
$$H(x,y) = \begin{bmatrix} S_{x2}(x,y) & S_{xy}(x,y) \\ S_{xy}(x,y) & S_{y2}(x,y) \end{bmatrix}$$

5. Compute the response of the detector at each pixel

$$R = Det(H) - k(Trace(H))^2$$

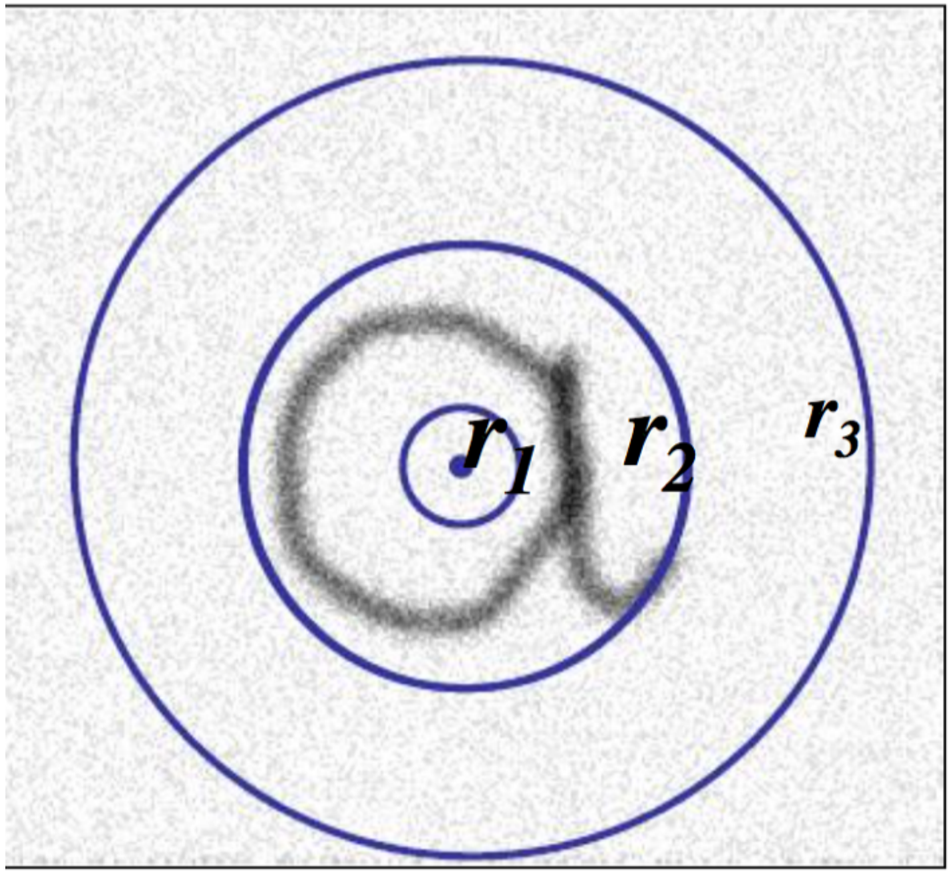
Characteristic Scale

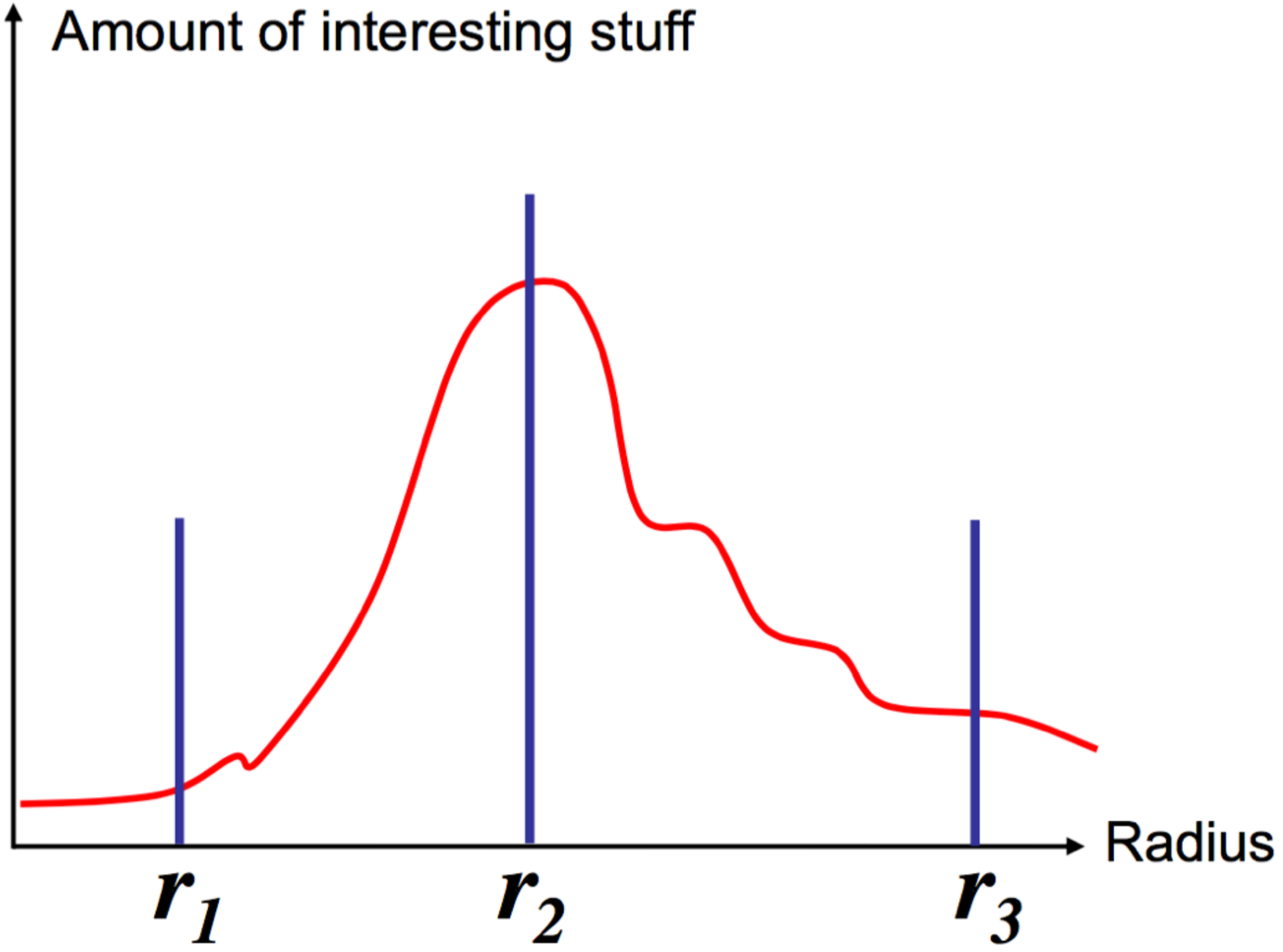
Scale Selection



- We need to decide what window size to use for computing the Harris matrix
- Equivalently, we need to choose the value of σ'
- The window size (or σ') must be consistent between different magnifications of the image

How to chose a neighborhood size?

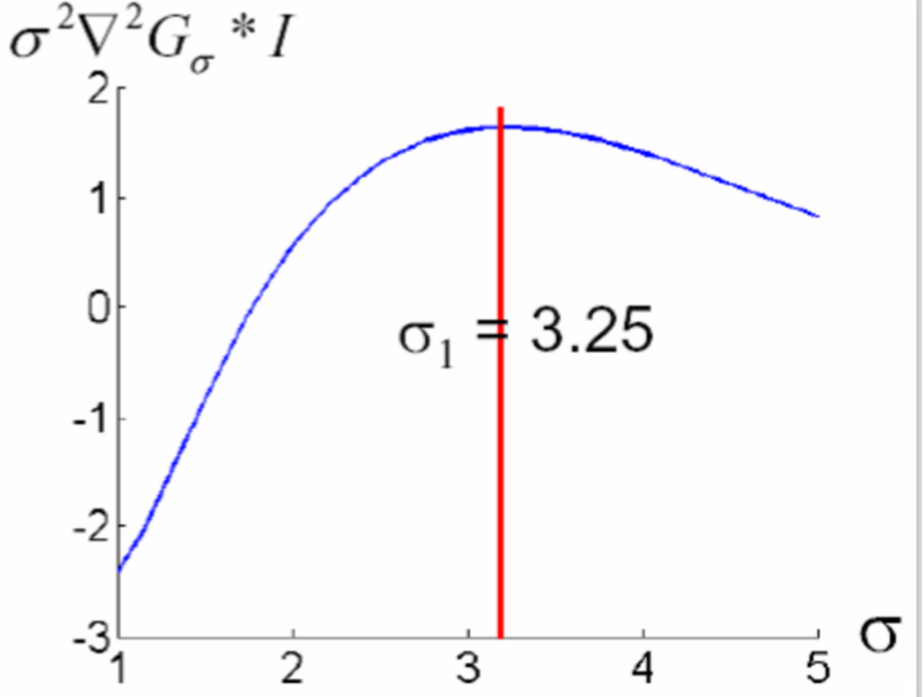


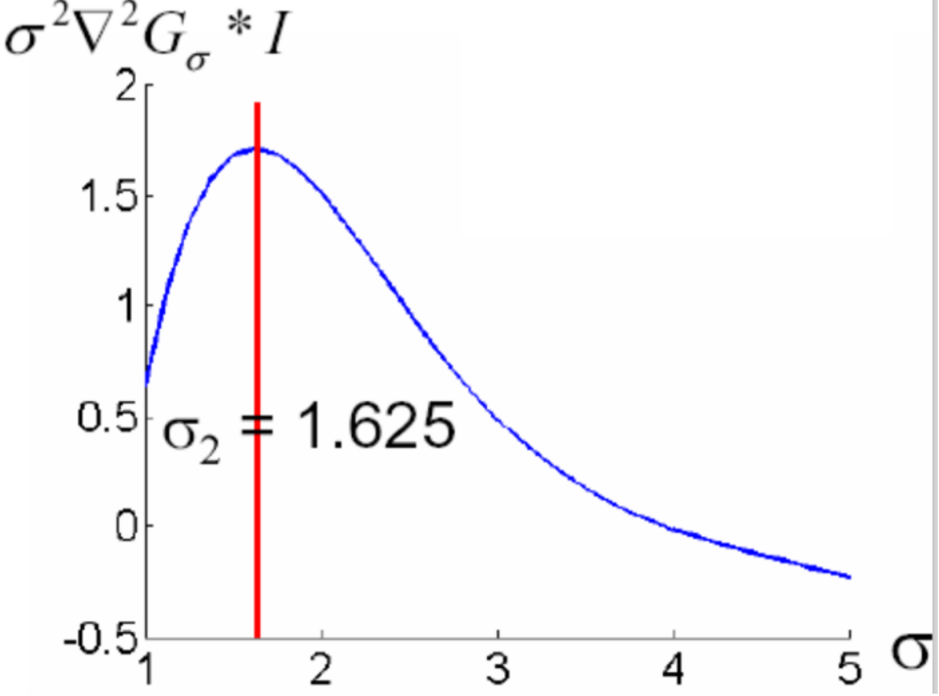


Best radius: Local extrema of function that measures amount of interesting stuff

X 2

$\sigma_1 = 2 \times \sigma_2$!!!





- Why did we use the normalized Laplacian in the previous example?
- Justification (and basis for most scale selection operations in computer vision):
- *Scale Selection Principle (T. Lindeberg):*

In the absence of other evidence, assume that a scale level, at which some (possibly non-linear) combination of normalized derivatives assumes a local maximum over scales, can be treated as reflecting a characteristic length of a corresponding structure in the data.

What are normalized derivatives?

$$\sigma^{n+m} \frac{\partial^{n+m} f}{\partial x^n \partial y^m}$$

Example using 2nd order derivatives

$$\sigma^2 \nabla^2 f = \sigma^2 \left(\frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \right)$$

Why do we Normalize?

First, let me try to give you some intuition of why you have to normalize by scale at all. As you go from finer to coarser scales you blur the image. That makes the intensity surface more and more smooth. That, in turn, means that the amplitude of image derivatives gets smaller as you go up the scale volume. This is a problem for finding interest points, because you are looking for local extrema over scale. Without normalization you will always get the maximum at the finest scale and the minimum at the coarsest scale, and that's not what you want.

So, image derivatives are attenuated as σ increases. In fact, the derivatives decrease exponentially as a function of σ. To compensate for that you have to normalize them by multiplying the n-th derivative by σ^n. Since the LoG is a combination of second derivatives, you have to multiply it by σ^2.

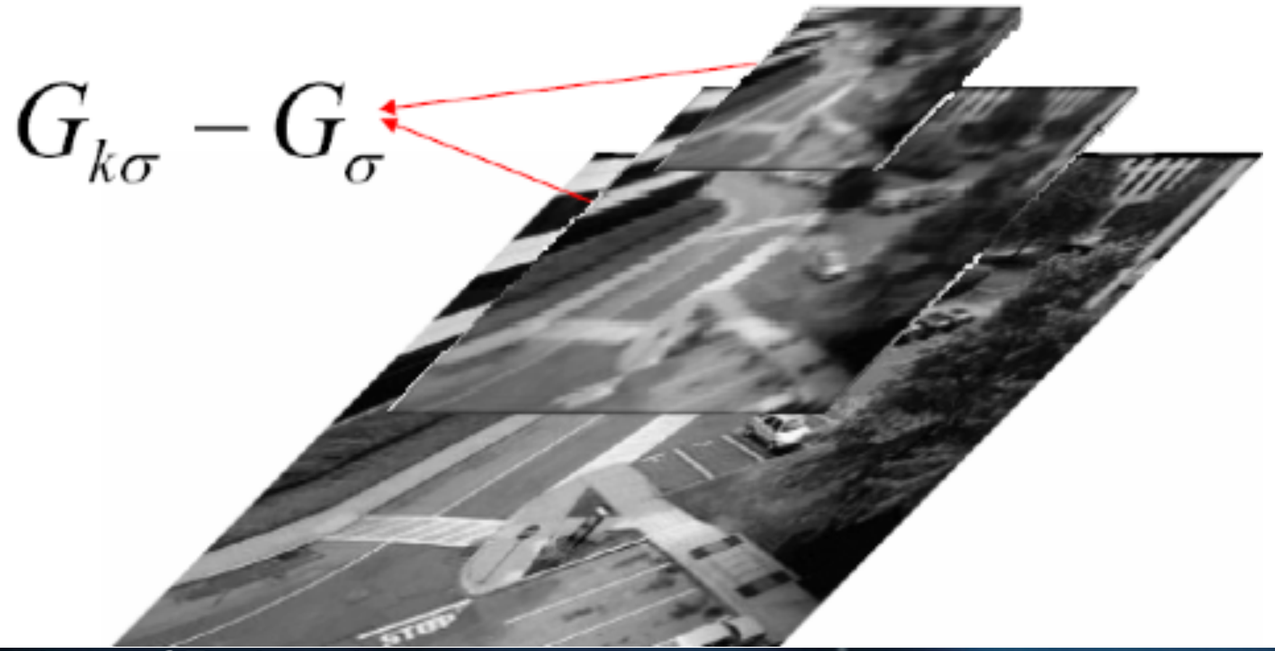
You can find the derivation and a better explanation of this in the paper by Toni Lindeberg.

http://math.stackexchange.com/questions/486303/normalized-laplacian-of-gaussian

$$\nabla^2_\sigma I = \nabla^2 G_\sigma * I$$

Constant independent of σ

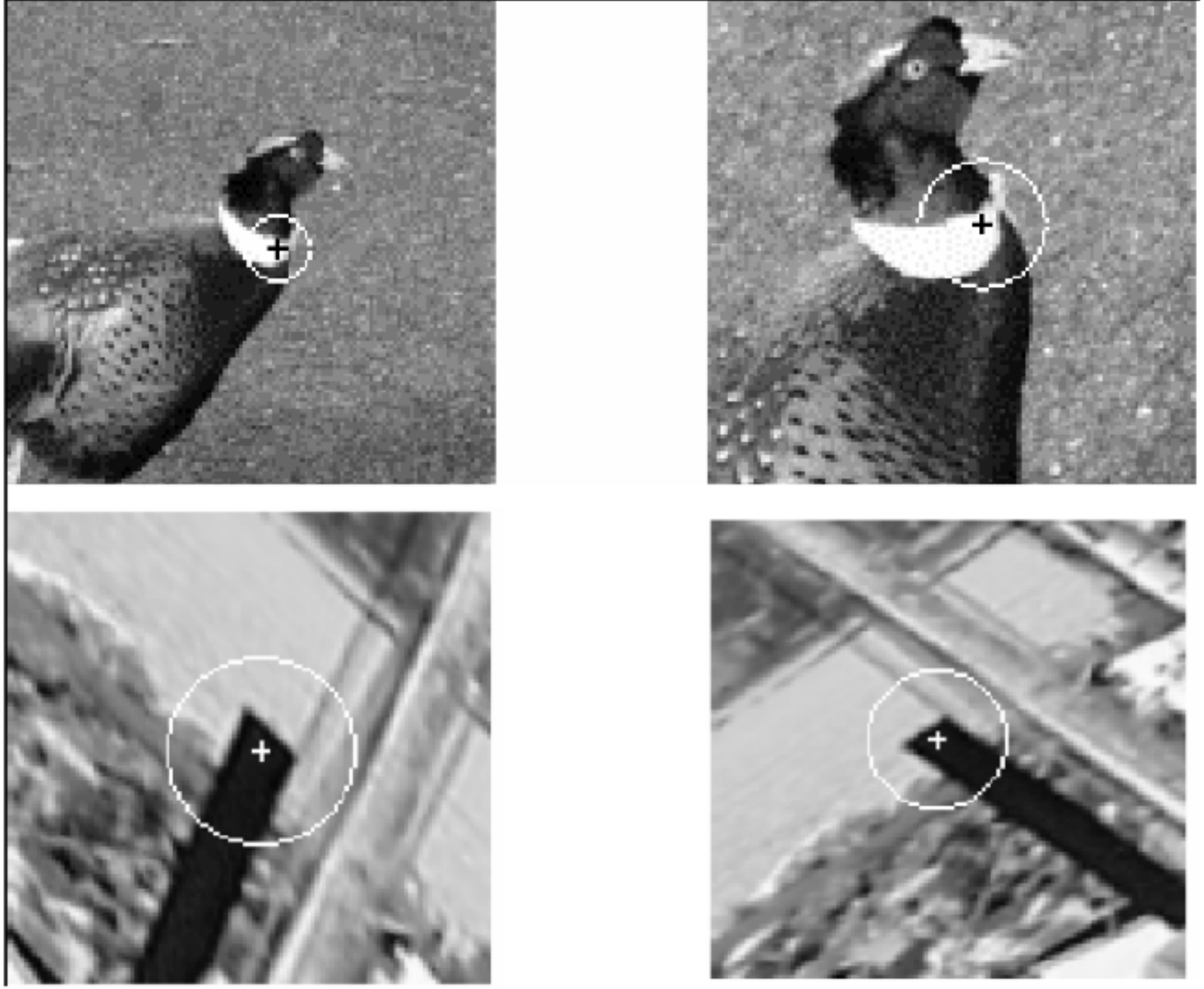
$$G_{k\sigma} - G_\sigma \approx (k-1)\sigma^2 \nabla^2 G_\sigma$$

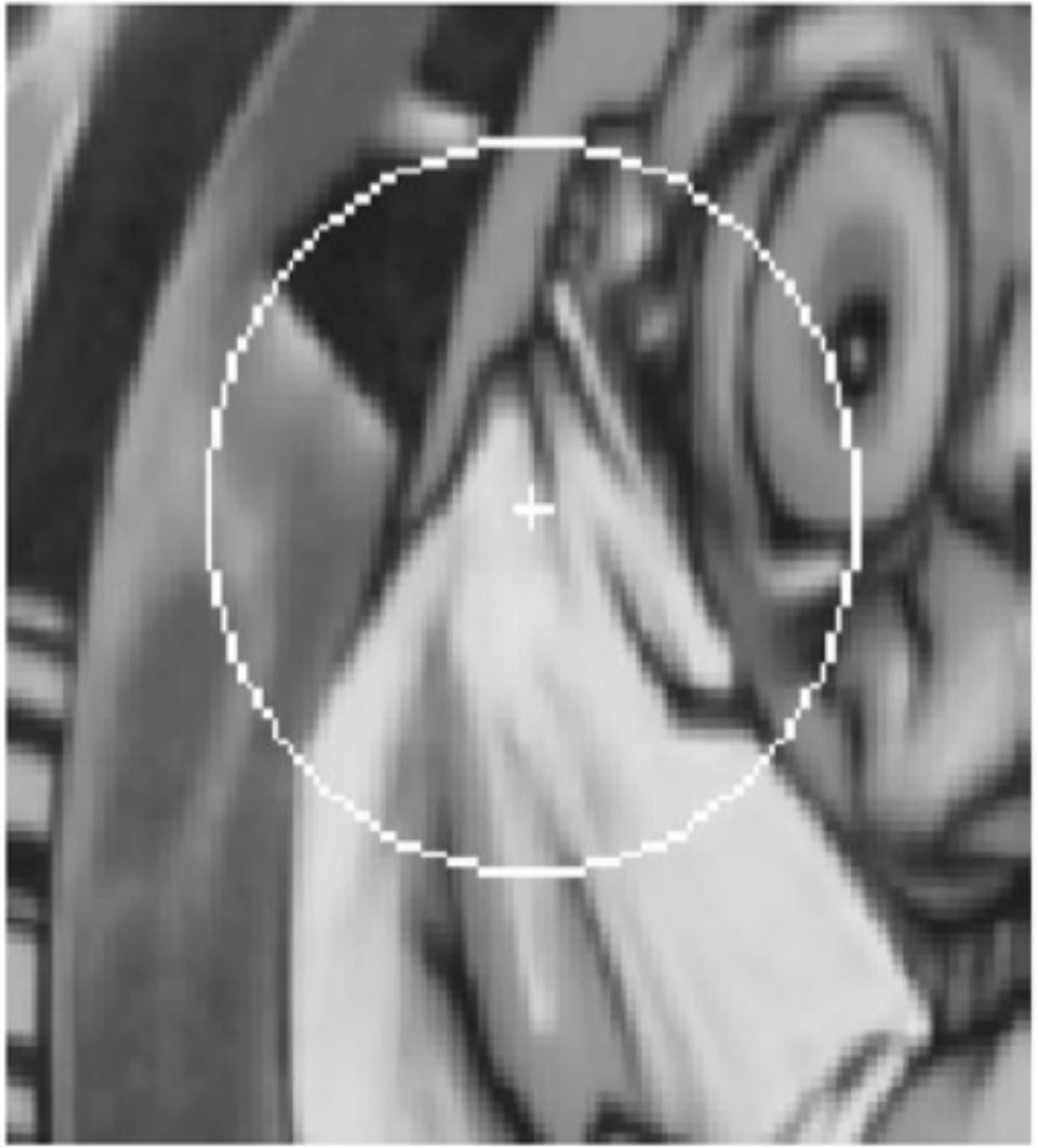


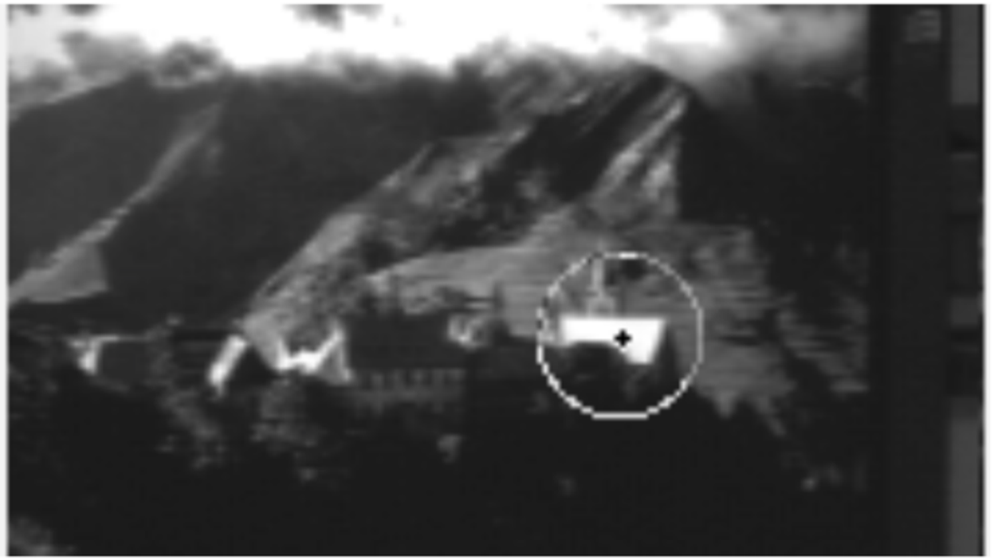
The Laplacian of a Gaussian can be approximated by the difference of two Gaussians.

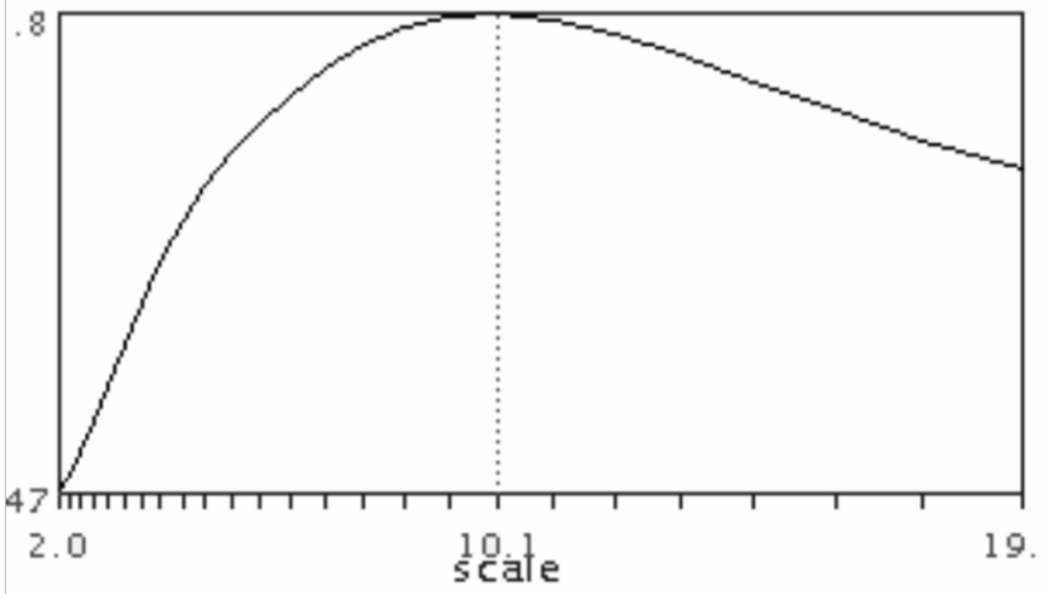
To compare the Laplacian at different scales we need to explain more carefully what the approximation is.

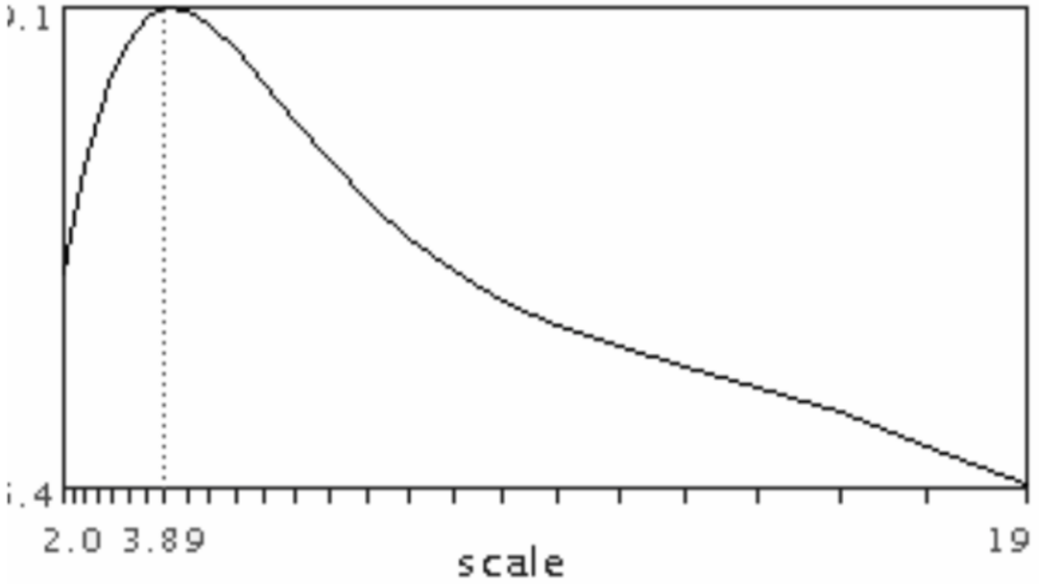
In practice: the scaled Laplacian can be computed by taking the difference between level in a Gaussian pyramid.









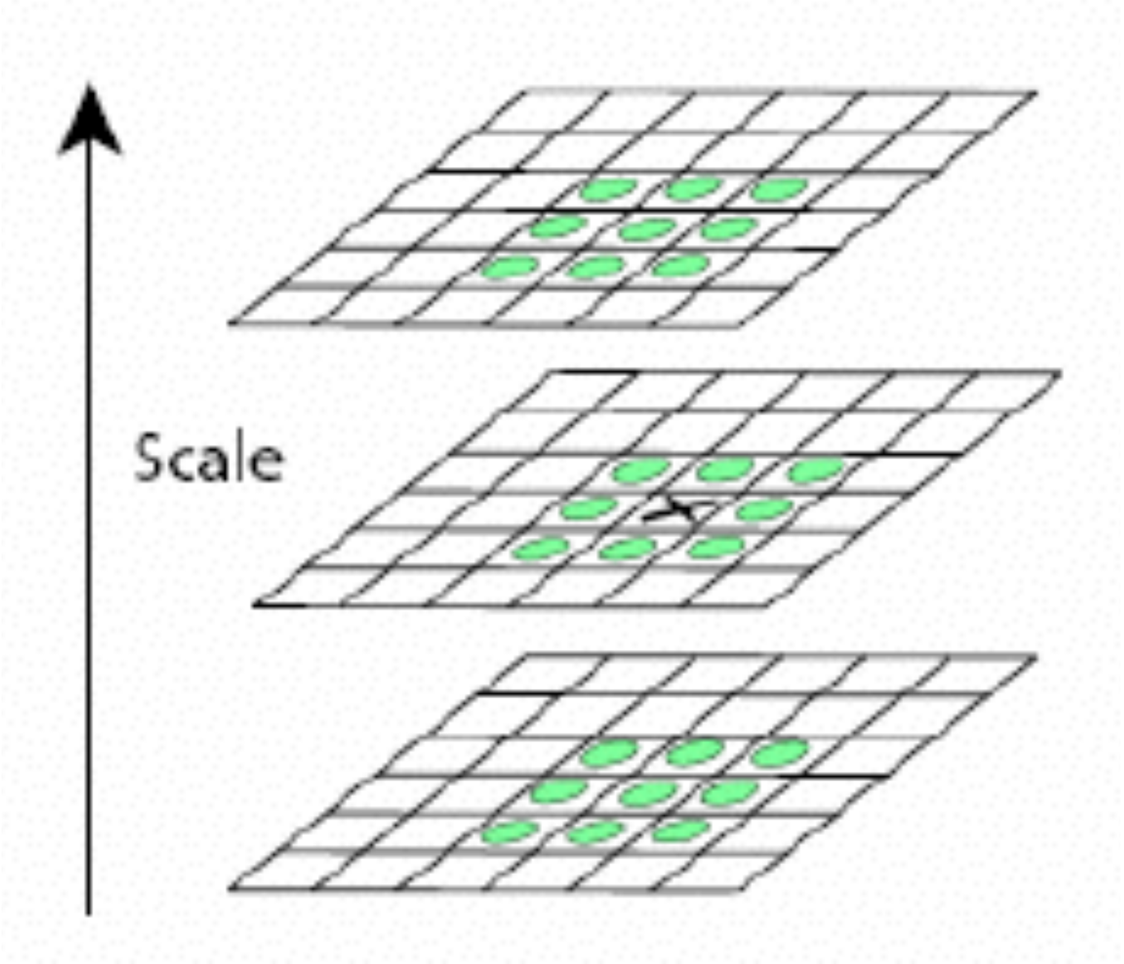


Laplacian

Interest points as “blob-like”:
Basic Laplacian detector =
Local maxima of characteristic scale

From Laplacian pyramid to detected points

- Find local extrema of $\nabla^2 G_\sigma$ over x,y, and σ (scale)



Combining the two: Harris-Laplace detector

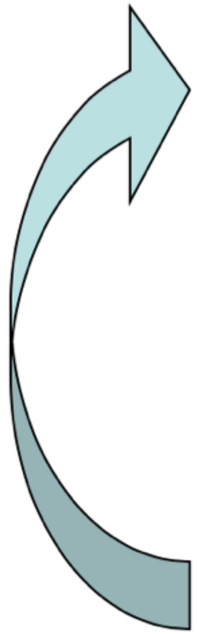
Combining Harris and Laplace: Harris-Laplace detector

$$\boldsymbol{H} = \boldsymbol{G}_{\sigma_I} * \begin{bmatrix} \boldsymbol{I}^2_{x,\sigma_D} & \boldsymbol{I}_{x,\sigma_D}\boldsymbol{I}_{y,\sigma_D} \\ \boldsymbol{I}_{x,\sigma_D}\boldsymbol{I}_{y,\sigma_D} & \boldsymbol{I}^2_{y,\sigma_D} \end{bmatrix} \quad \boldsymbol{R} = \boldsymbol{Det}(\boldsymbol{H}) - \boldsymbol{k}(\boldsymbol{Trace}(\boldsymbol{H}))^2$$

- The location x of the points detected by Harris is not scale invariant → Depends on the choice of σ_I and σ_D.
- Reduce to one parameter: $\sigma_D = s\sigma_I$ ($s = 0.7$)
- The Laplacian trick gives us a good σ but not where the interest point is.
- Chicken and egg problem:
  - If we knew x we could estimate σ
  - If we knew σ we could find x

Harris-Laplace: Algorithm *summary*

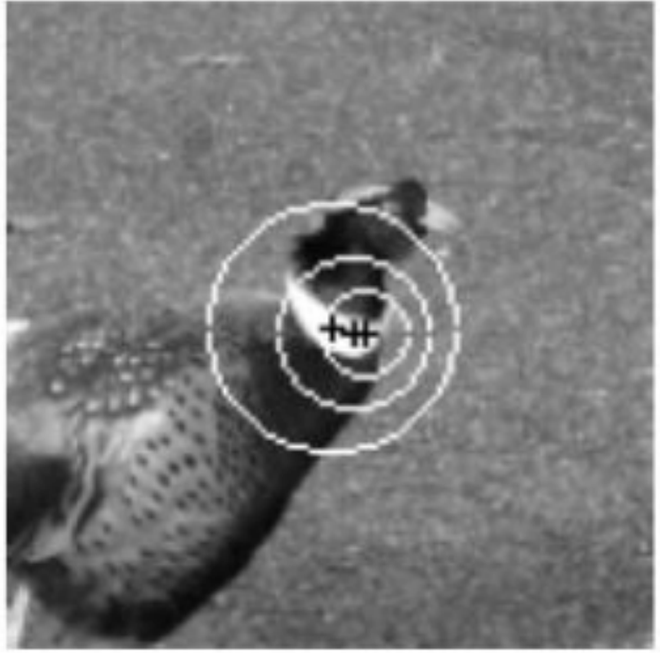
1. Try Harris at different scales and report initial points and associated scales
   - Try different σ_l of the form $k^n\sigma_o$ $(k=\sqrt{2})$
   - Report points with large R

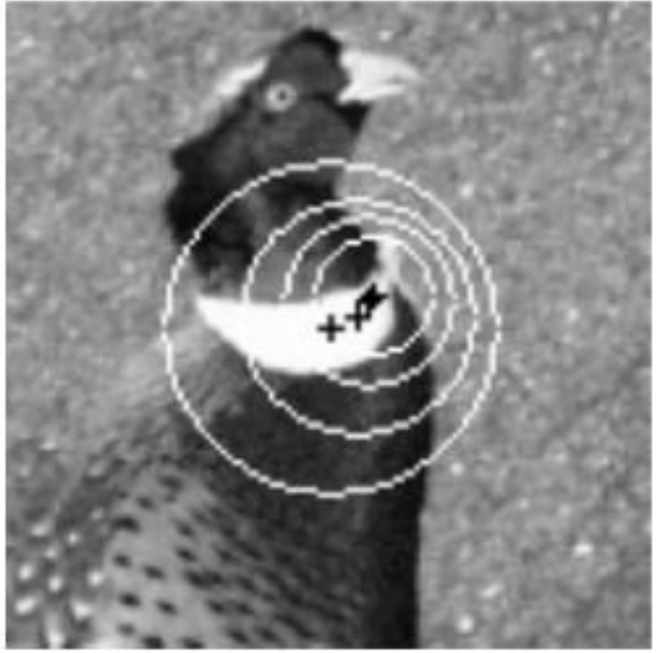


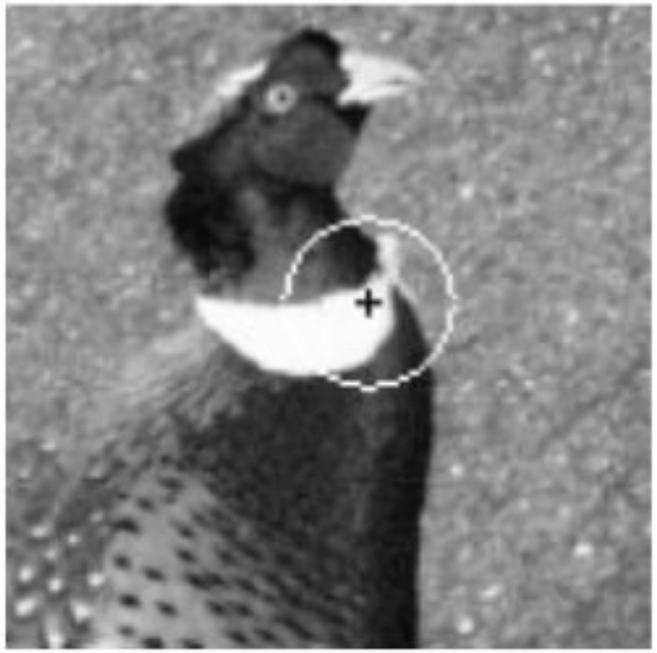
2. For each detected point x
   - Estimate characteristic scale σ_c as maximum of

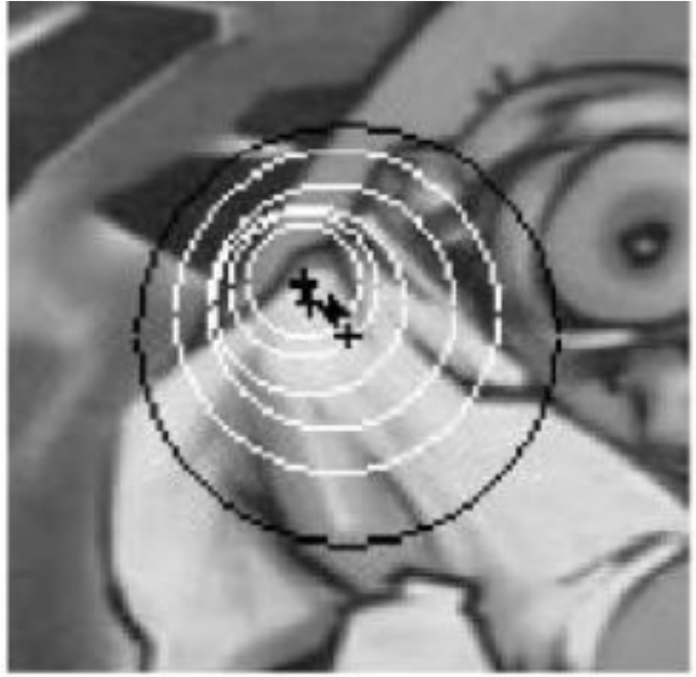
$$\sigma^2\nabla^2\boldsymbol{G}_\sigma$$

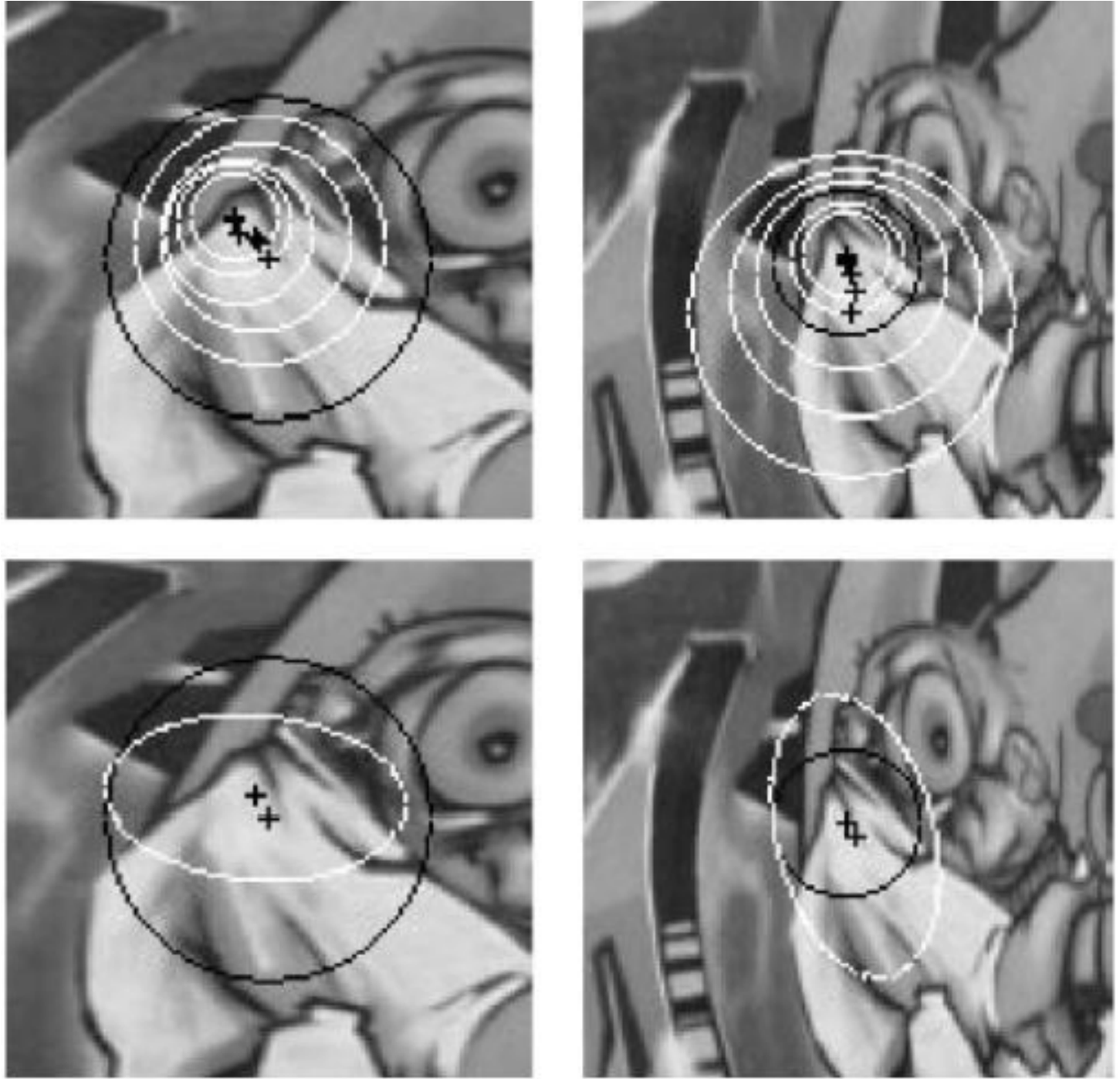
   - Find the point x' with the maximum of R in a 8x8 neighborhood of x by using the new scale $\sigma_l = \sigma_c$
   - Replace x by x'











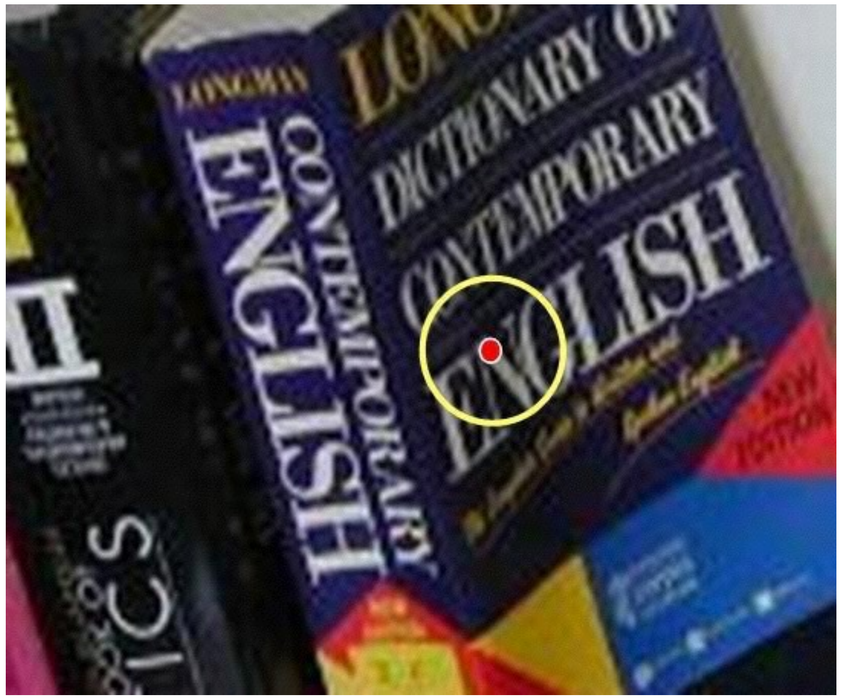
What about viewpoint changes?!

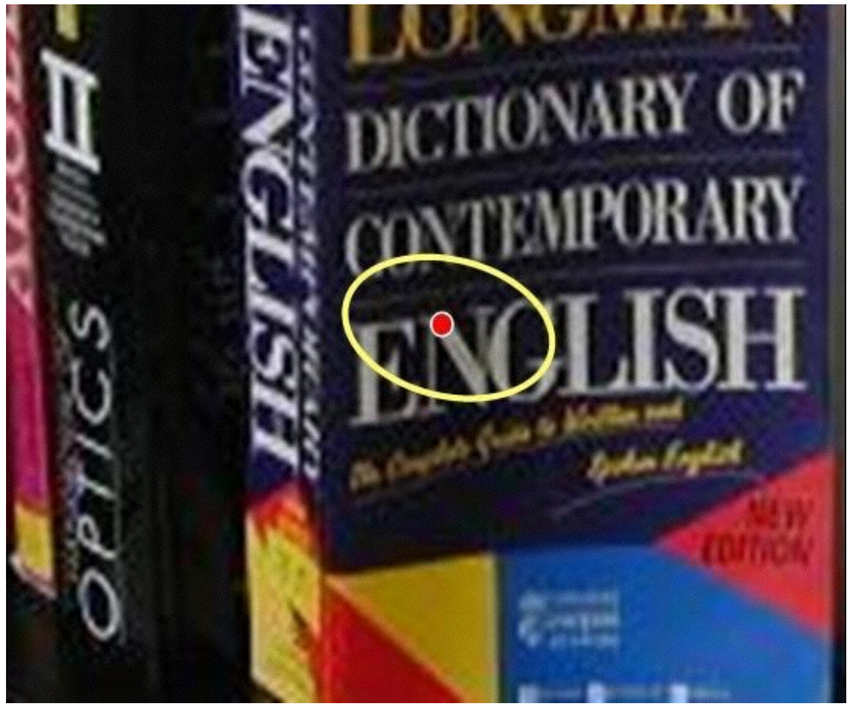
Affine Invariant Detection (overview)

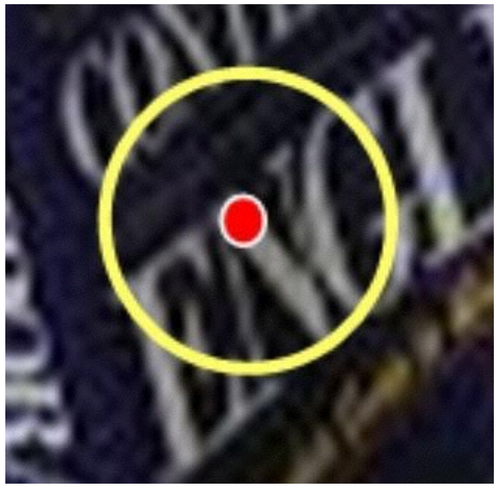
- Need to define a richer description of “neighborhood” or scale
- Use directional derivatives instead: σ_I replaced by Σ_I, σ_D replaced by Σ_D
- Σ_I represent an elliptical “neighborhood” instead of a circular one
- More degrees of freedom to search through but conceptually similar algorithm:

- Assume $\Sigma_D = s\Sigma_I$
- Find x’s with initial selections of Σ_I
- Iterate:
  - Re-estimate “scale” Σ_I
  - Adjust the location of x based on new Σ_I

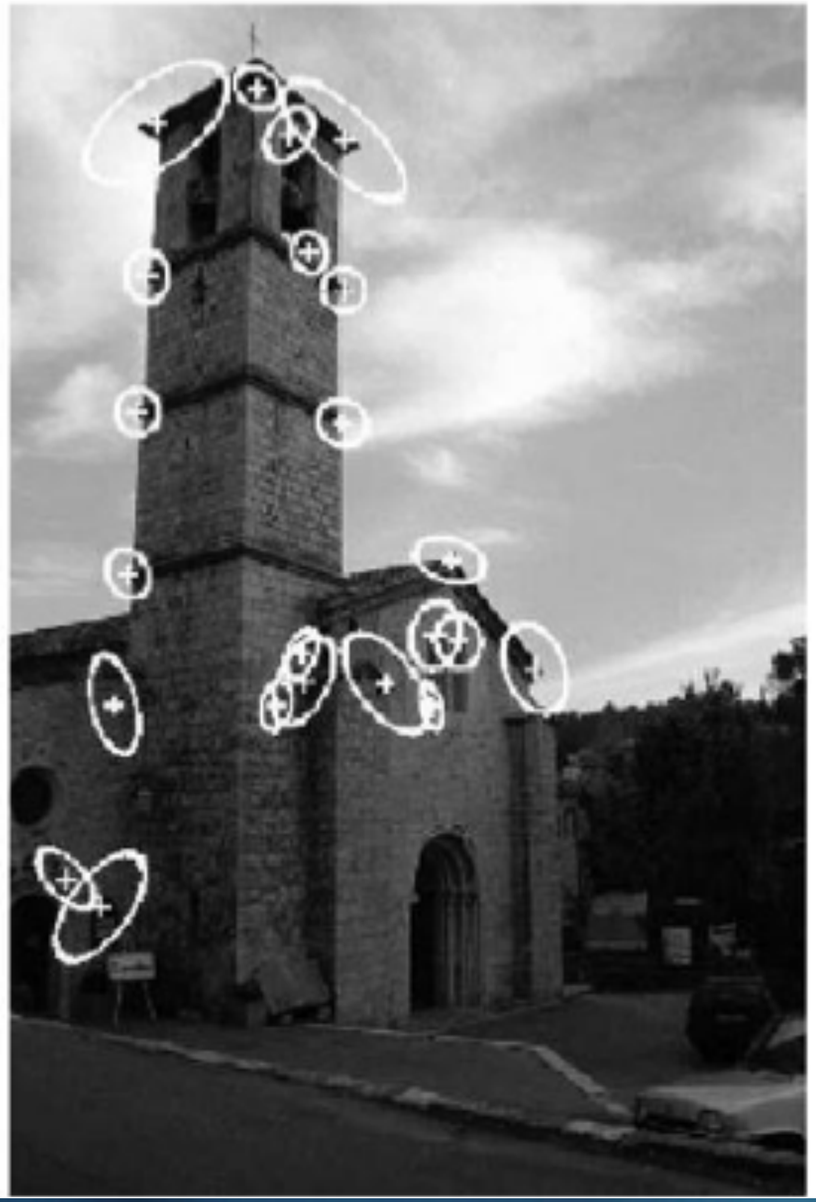
Affine Invariance







Application - Finding Correspondences



Example from Mikolajczyk and Schmid 2004

Scale: 4.9 Rotation: 19^{o}

Example from Mikolajczyk and Schmid 2004

Scale: 4.9 Rotation: 19^{o}

Example from Mikolajczyk and Schmid 2004

Scale: 1.7 Rotation: 50°