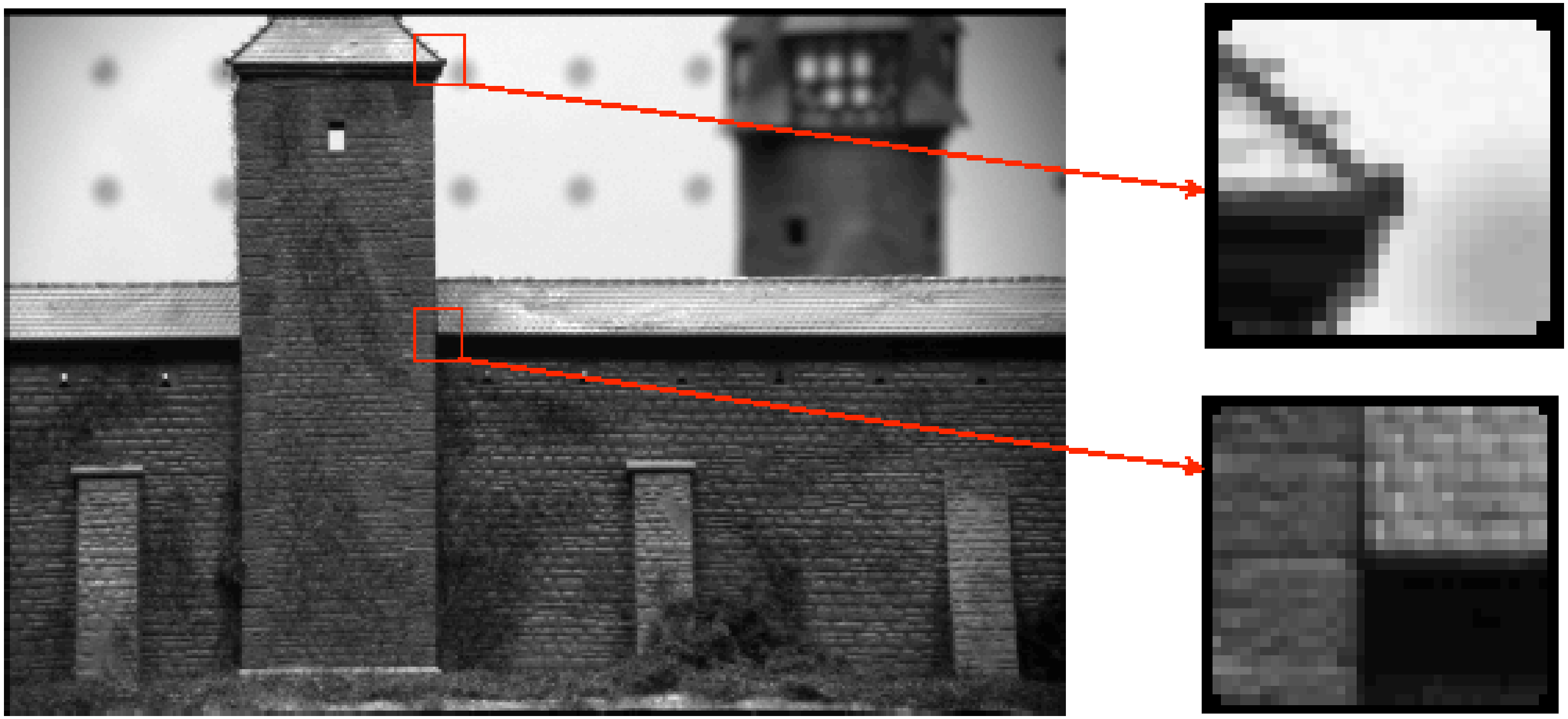
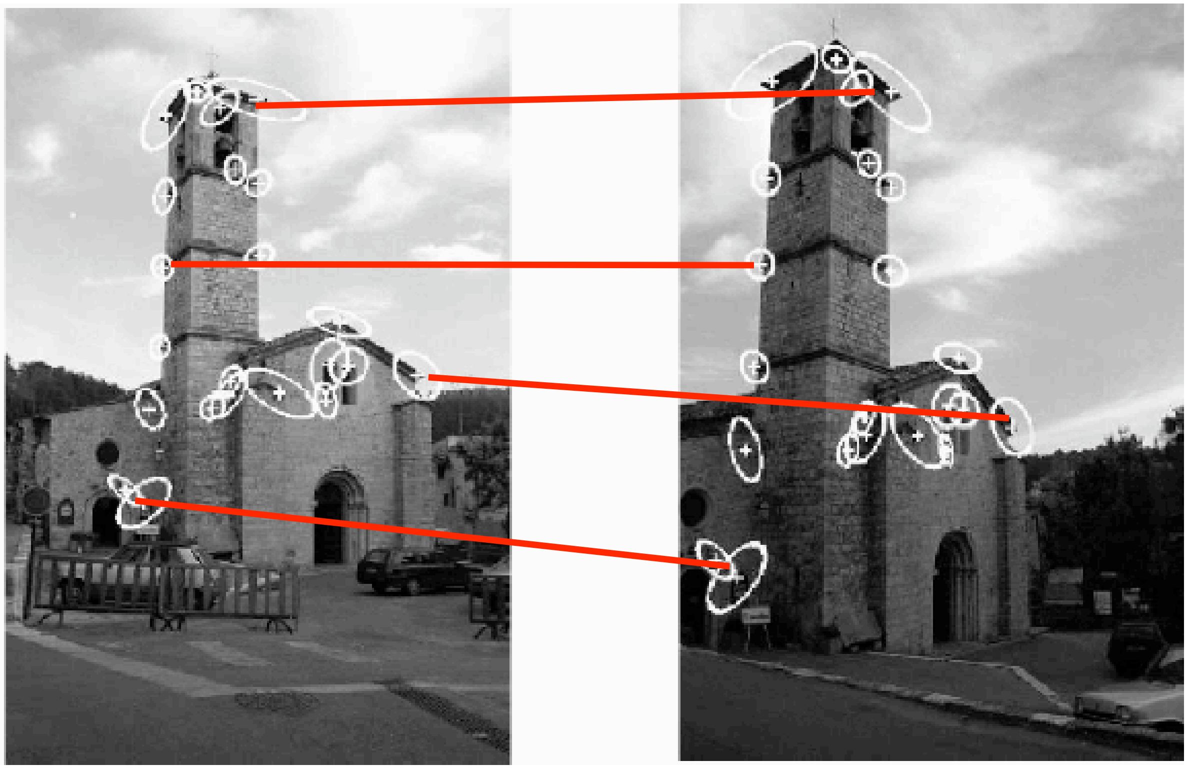
CORNER DETECTION

Intuition of a Corner



- Intuitively they are “junctions of contours”.

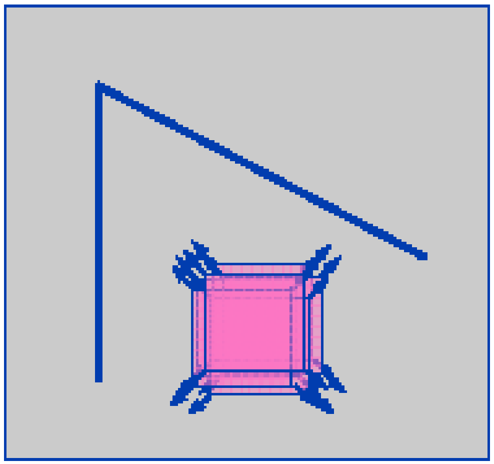
Why do we care about corners?



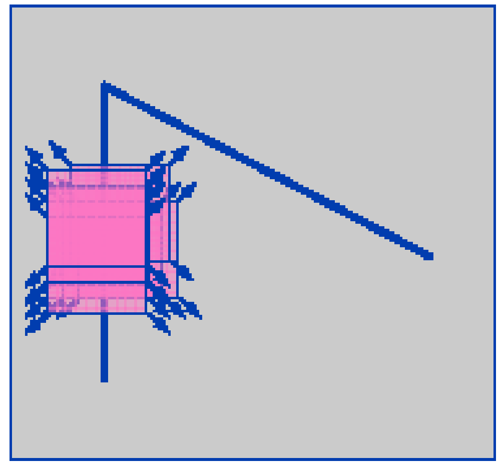
- They can help us to match two images.
- Useful to recover 3D structure, create panoramic images, tracking, and more!
- Good features to match!

From http://www.cse.psu.edu/~rcollins/CSE486/

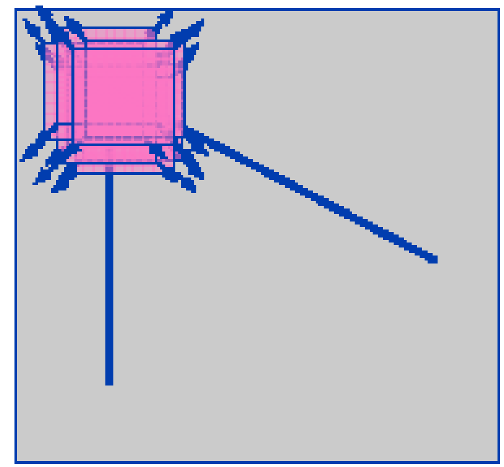
Harris Corner Detection



“flat” region:
no change in
all directions



“edge”:
no change along
the edge direction



“corner”:
significant change
in all directions

Harris corner detector gives a mathematical approach for determining which case holds.

From http://www.cse.psu.edu/~rcollins/CSE486/

Harris Corner Detection (cont'd)

$$E(u,v) = \sum_{x,y} \underbrace{w(x,y)}_{\text{Window (e.g., Gaussian)}} \left(\underbrace{I(x+u,y+v)}_{\text{Shifted Intensity}} - \underbrace{I(u,v)}_{\text{Intensity}}\right)^2$$

- For nearly constant patches, the difference is small.
- For distinctive patches, the difference is large.

From http://www.cse.psu.edu/~rcollins/CSE486/

Taylor series for 2D functions

- First order approximation:

$$f(x+u, y+v) \approx f(x,y) + u f_x(x,y) + v f_y(x,y)$$

Harris Corner Detection (cont'd)

Taylor approximation

$$\sum_{x,y} \left(I(x+u, y+v) - I(u,v)\right)^2 \approx \sum \left(I(x,y) + uI_u(x,y) + vI_v(x,y) - I(x,y)\right)^2$$

$$= \sum_{x,y} u^2 I_u^2 + 2uv + v^2 Iv^2$$

$$= \sum_{x,y} \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} Iu^2 & I_u I_v \\ I_v I_u & I_v^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

$$= \begin{bmatrix} u & v \end{bmatrix} \left(\sum_{x,y} \begin{bmatrix} Iu^2 & I_u I_v \\ I_v I_u & I_v^2 \end{bmatrix} \right) \begin{bmatrix} u \\ v \end{bmatrix}$$

From http://www.cse.psu.edu/~rcollins/CSE486/

Harris Corner Detection (cont'd)

- Rewriting

$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix}$$

- The matrix M (2x2) is

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_u^2 & I_u I_v \\ I_v I_u & I_v^2 \end{bmatrix}$$

From http://www.cse.psu.edu/~rcollins/CSE486/

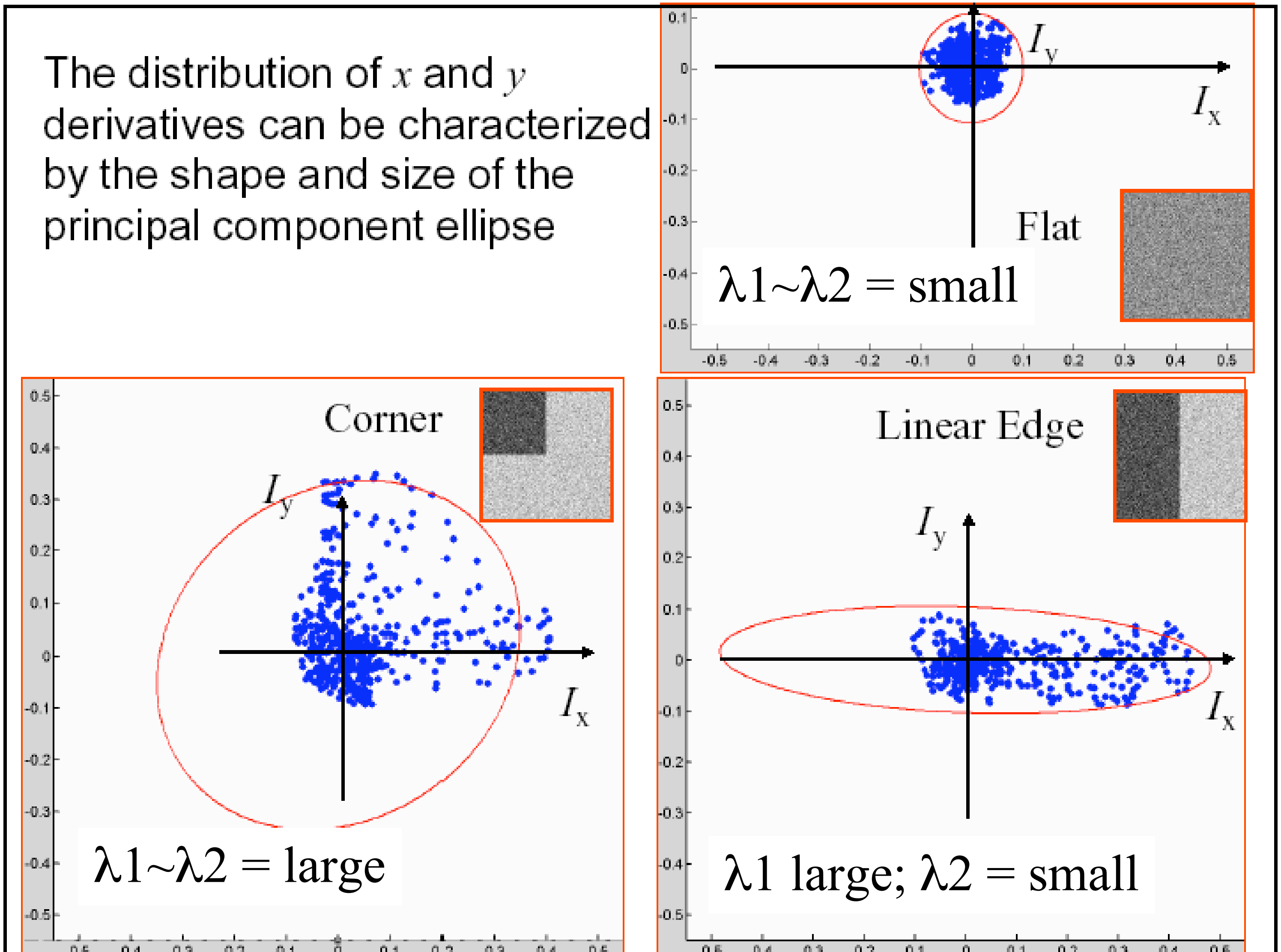
Intuition about Harris

- Assume gradients are 2D points (I_x, I_y)
- Fit an ellipse to the set of points via scatter matrix
- Analyze ellipse parameters to determine if we detected a corner

From http://www.cse.psu.edu/~rcollins/CSE486/

Intuition about Harris (visualization)

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



From http://www.cse.psu.edu/~rcollins/CSE486/

Classification via Eigenvalues

- If both eigenvalues are large => corner!
- If single eigenvalue is large => edge.
- If both eigenvalues are small => flat region
- Computing eigenvalue is expensive!

Measure of corner response

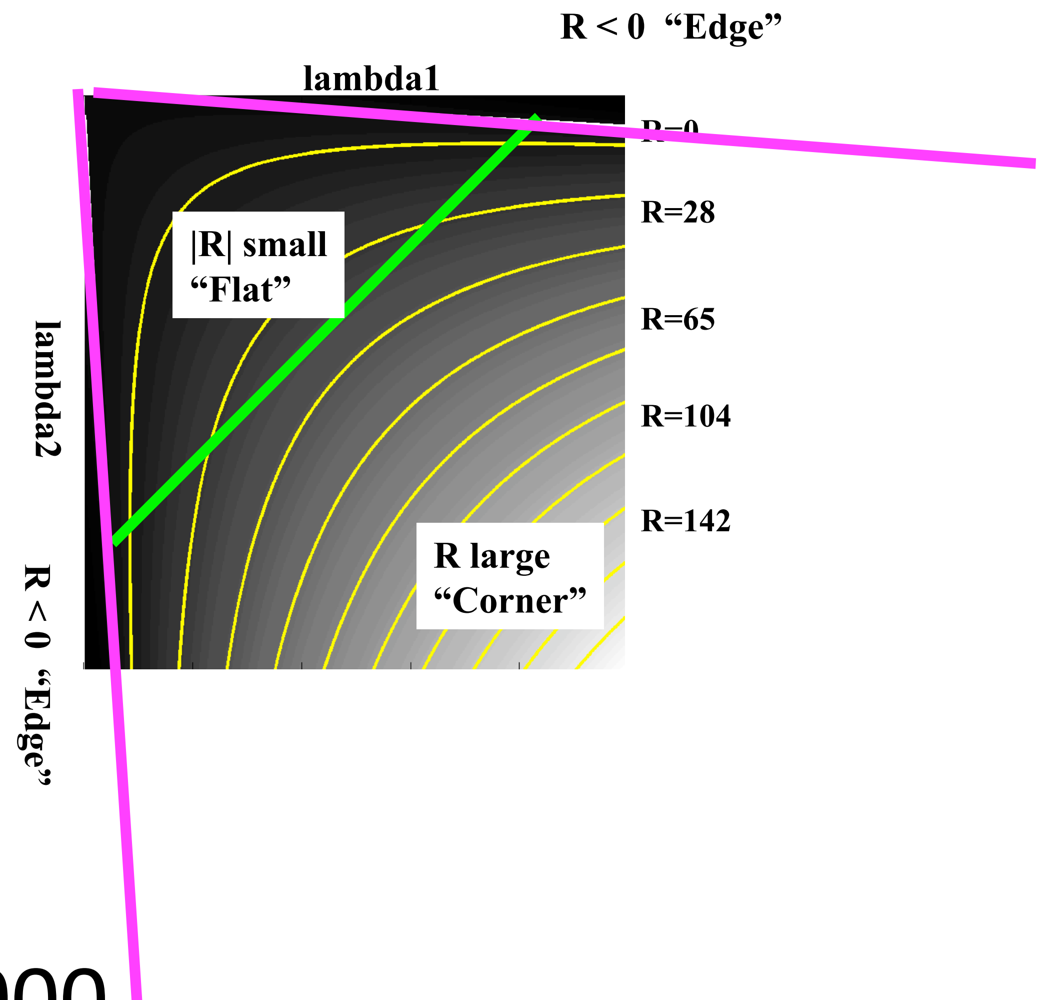
- Corner response:

$$R = \det M - k \ \mathrm{trace}(M)^2$$

- Constant k in the range (0.02 – 0.06)
- Determinant is the product of the eigenvalues
- Trace is the sum of the eigenvalues

Corner response map

- R depends only on eigenvalues of M
- R is large for a corner
- R is negative with large magnitude for an edge
- $|R|$ is small for a flat region



Usually R > 10000

From http://www.cse.psu.edu/~rcollins/CSE486/

Harris corner detector algorithm

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

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

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

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

6. Threshold on value of R. Compute nonmax suppression.

From http://www.cse.psu.edu/~rcollins/CSE486/

Shi-Tomasi corner test

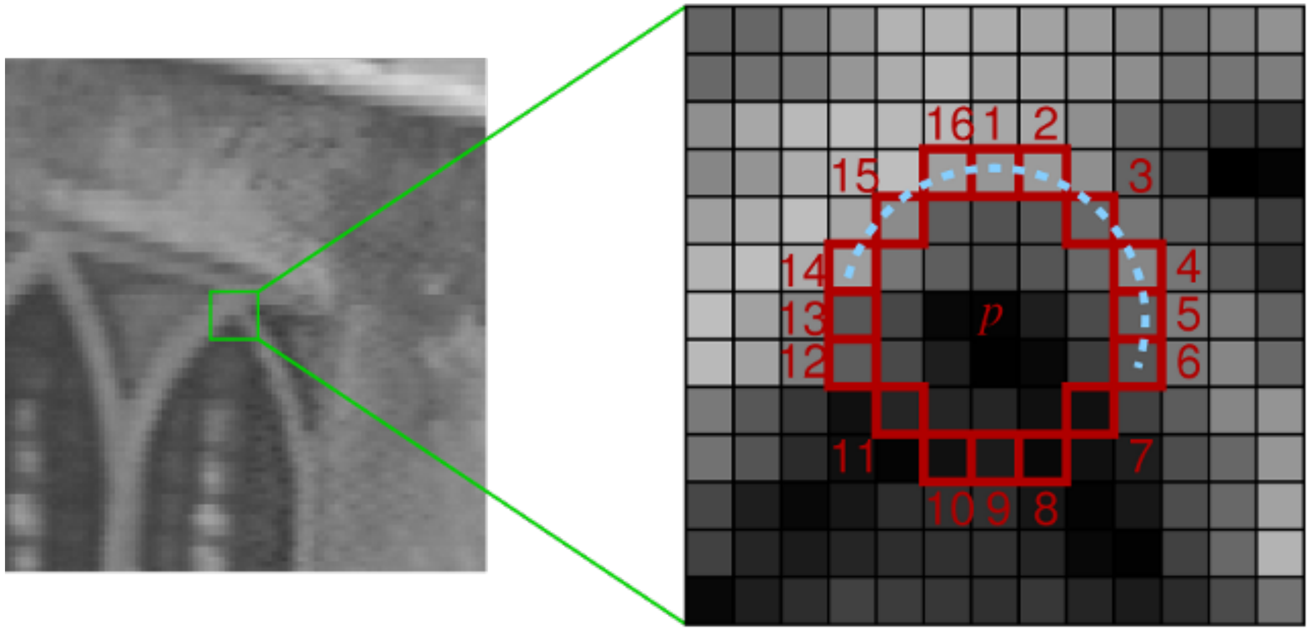
$$\min(\lambda_1, \lambda_2) > \lambda$$

Threshold

- Accept as corner if above threshold.

J. Shi and C. Tomasi. Good features to track. CVPR 1994

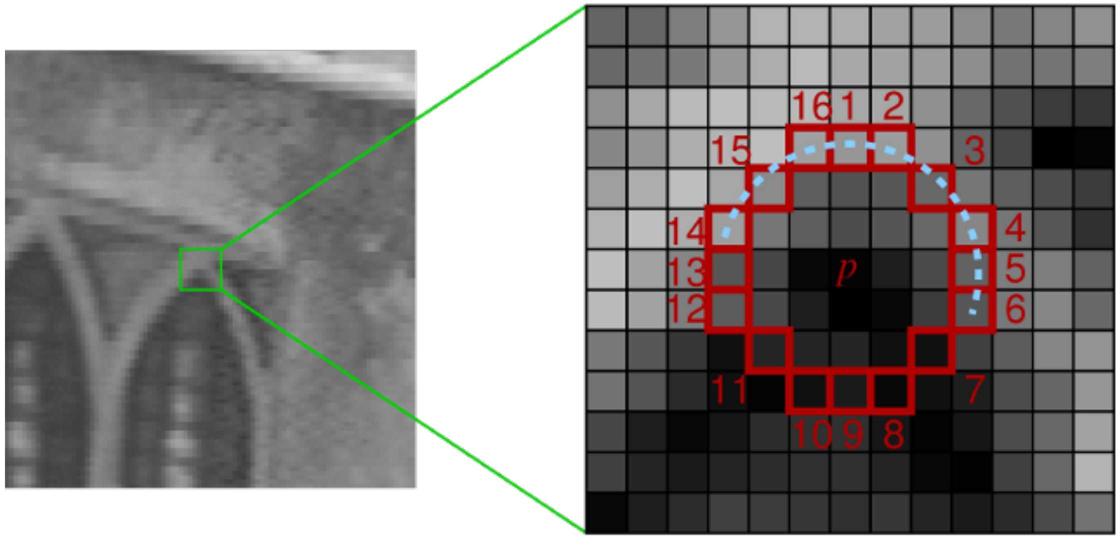
FAST corners: Features from Accelerated Segment Test



- Uses a Bresenham circle of radius 3 (16 pixels).
- Defines a fast test to classify pixels as corners or non corners.

E. Rosten and T. Drummond. Machine learning for high-speed corner detection. ECCV'06

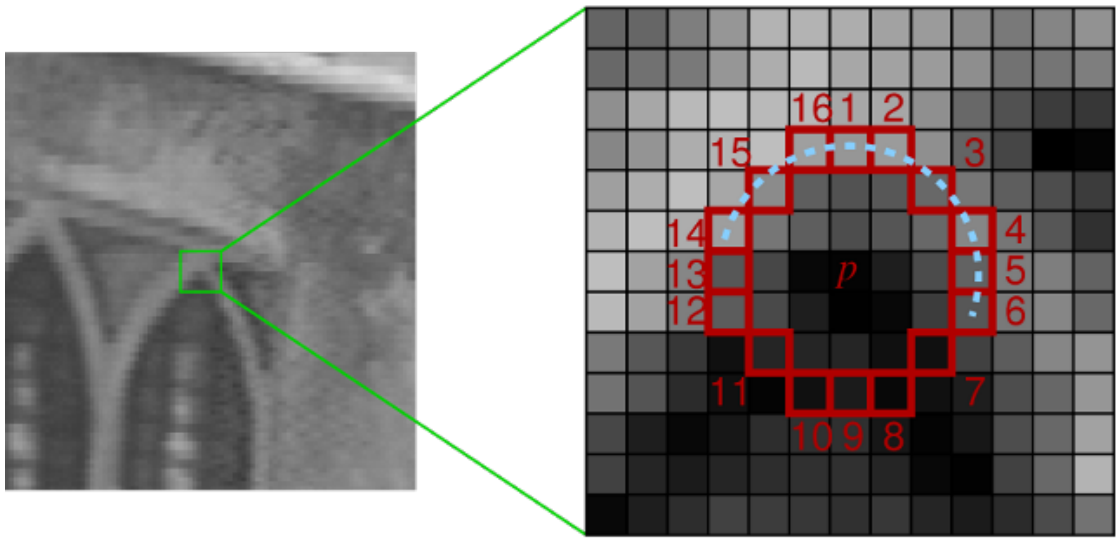
FAST corners: Features from Accelerated Segment Test



Condition:

1. If N (typically 12) contiguous pixels are brighter/darker than the current intensity +/- threshold, then classify as corner.

High-speed test of non-corners



- Examine pixels 1, 5, 9, 13.
- If current pixel is a corner, then at least 3 pixels examinations must be brighter/darker than Ip + t / Ip – t, respectively.
- If this is not satisfied, reject.

Weaknesses of the high-speed test.

1. The high-speed test does not generalize well for $n < 12$.
2. The choice and ordering of the fast test pixels contains implicit assumptions about the distribution of feature appearance.
3. Knowledge from the first 4 tests is discarded.
4. Multiple features are detected adjacent to one another.

Using machine learning to address the weaknesses

- Detect corners to generate a training set.
- For every pixel I_x in the circle (1-16) of the training set, label them as:
  - Darker if $I_x <= I_p - t$
  - Similar if $I_p - t <= I_x <= I_p + t$
  - Brighter if $I_x <= I_p + t$
- Train a decision tree for every pixel in the circle.
- The decision tree can be implemented efficiently in C/C++.

BLOB DETECTION

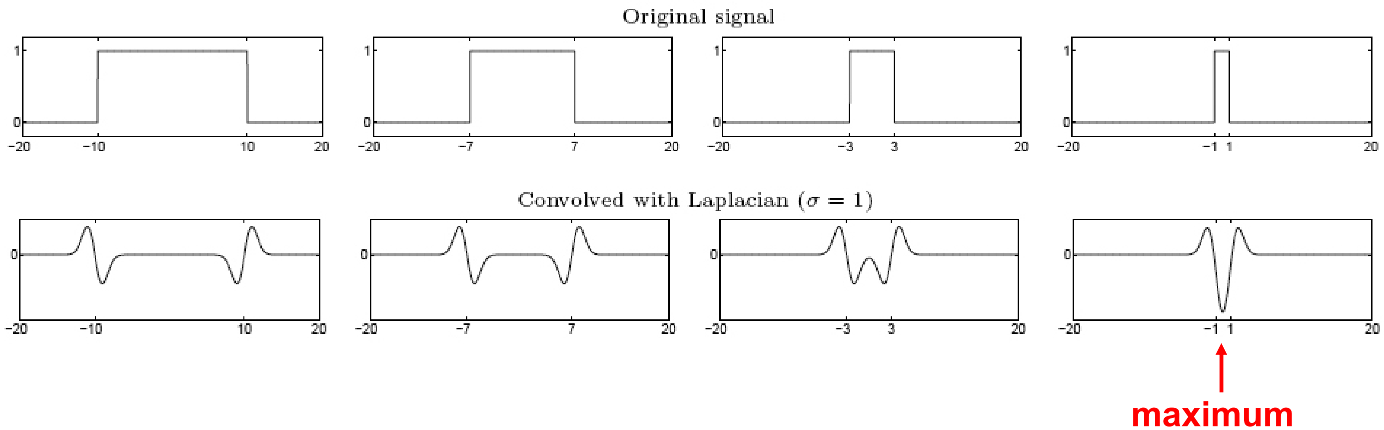
Intuition of a blob

A “radially” contrasting region

Image from http://www.cs.utah.edu/~manasi/

Blobs in 1D

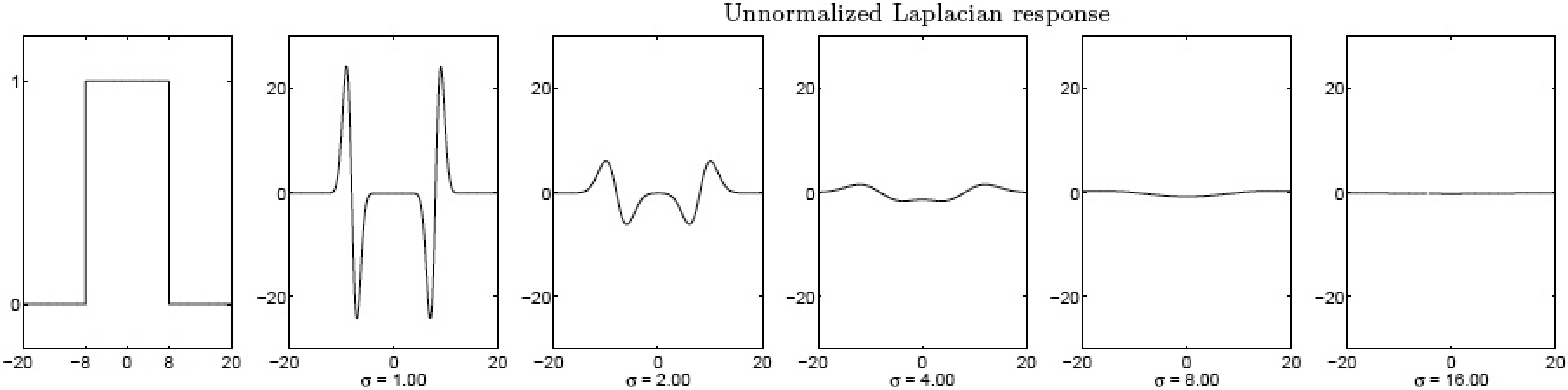
- Edge = ripple
- Blob = superposition of two ripples



- The magnitude of the Laplacian response will achieve a maximum at the center of the blob.
- This happens when the scale of the Laplacian filter (i.e., its sigma parameter) matches the scale of the blob.
- Another way of thinking about this is when the similarity, measured by convolution, between the signal and the Laplacian is maximum.

Image from http://www.cs.unc.edu/~lazebnik/spring11/lec08_blob.pdf

Laplacian response decays as scale increases



original signal
(radius=8)

increasing σ

Image from http://www.cs.unc.edu/~lazebnik/spring11/lec08_blob.pdf

Scale normalization

- To keep the responses the same (scale-invariant), we must multiply Gaussian derivative by **σ.**
- Laplacian is the second derivative of a Gaussian, so we must multiply by $\boldsymbol{\sigma}^{2}$.

Effect on scale normalization

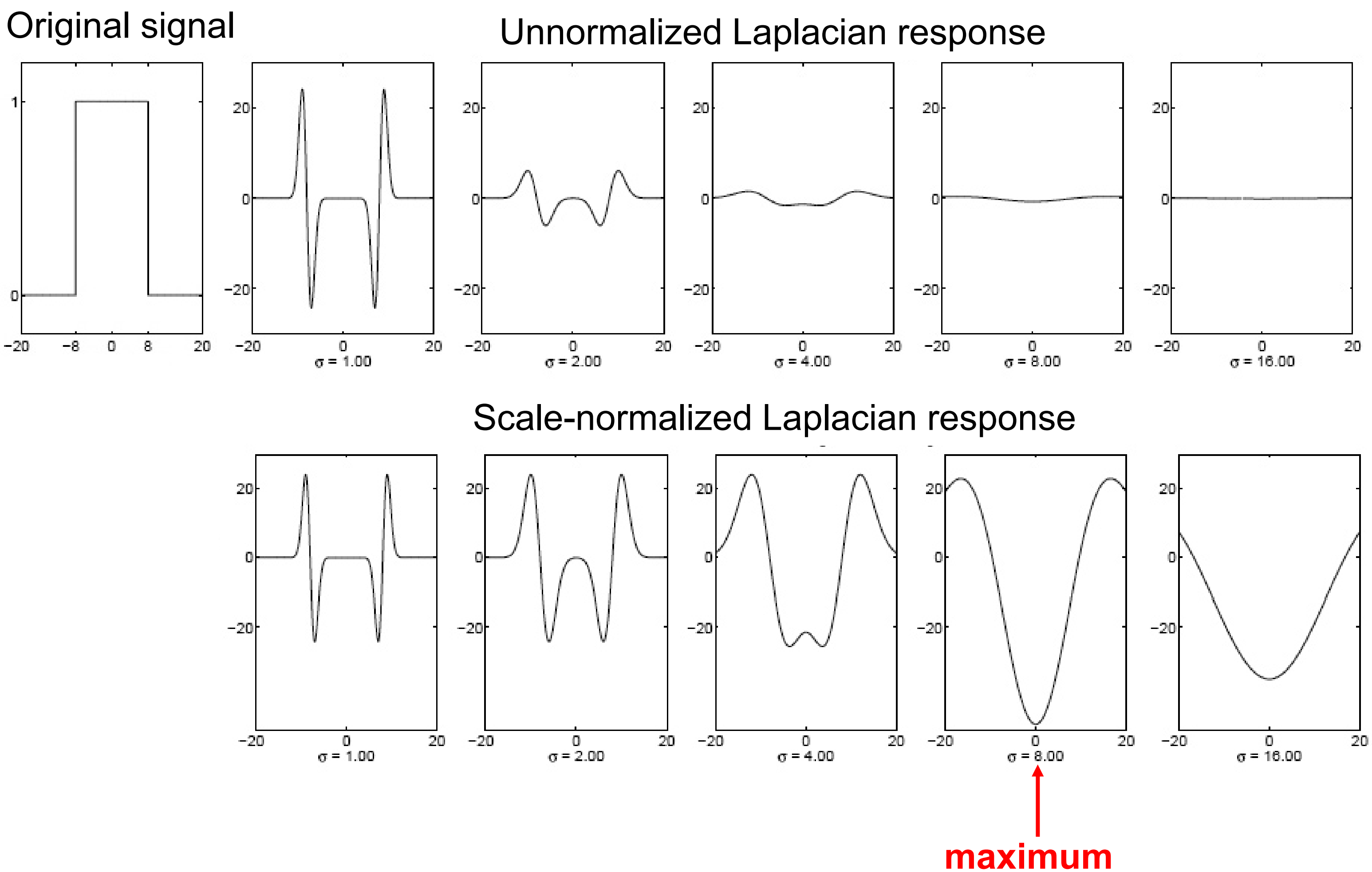
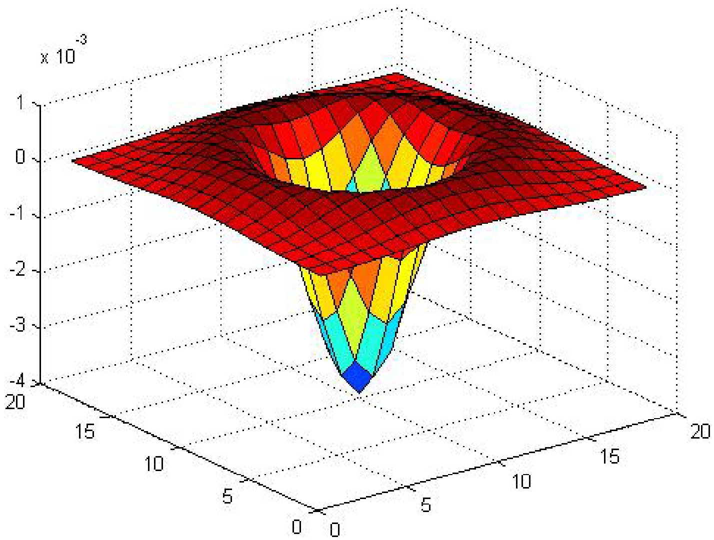
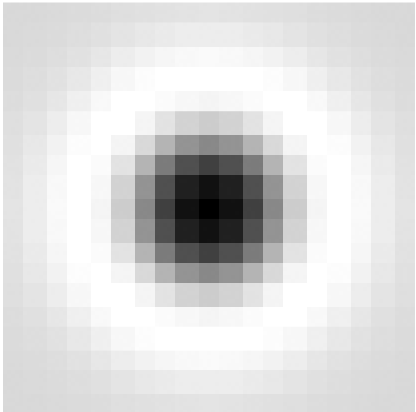


Image from http://www.cs.unc.edu/~lazebnik/spring11/lec08_blob.pdf

Blob detection in 2D

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



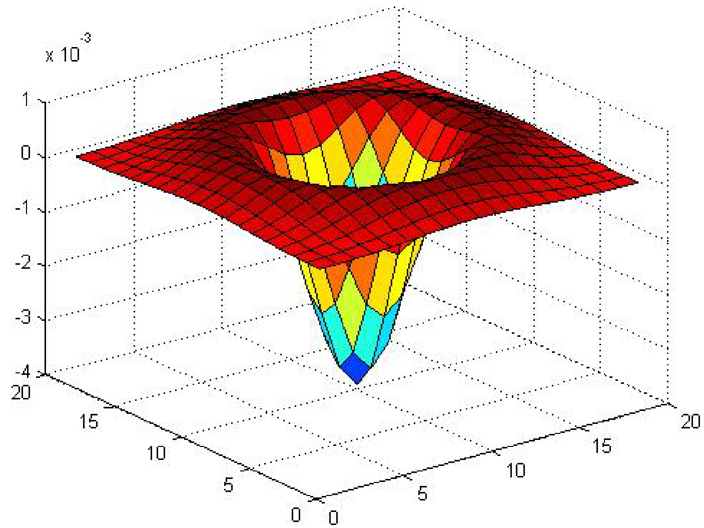


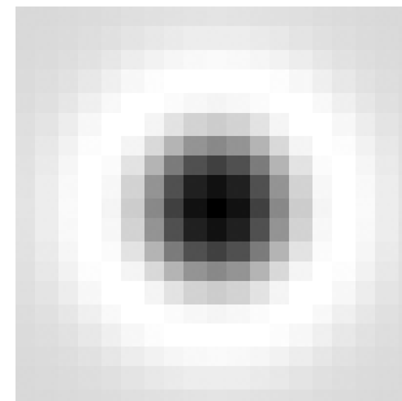
$$\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}$$

Image from http://www.cs.unc.edu/~lazebnik/spring11/lec08_blob.pdf

Blob detection in 2D normalized

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





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

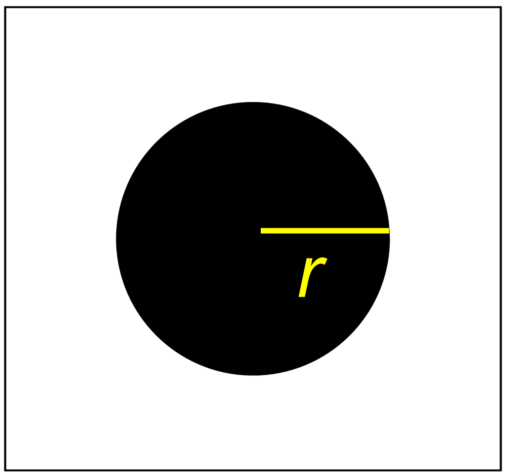
Image from http://www.cs.unc.edu/~lazebnik/spring11/lec08_blob.pdf

Scale selection

- What is the σ parameter of the LoG so that we can detect a circle of radius r?
- To get maximum response, the zeros of the LoG have to be aligned with the circle. This happens at

$$\sigma = \frac{r}{\sqrt{2}}$$

Scale selection



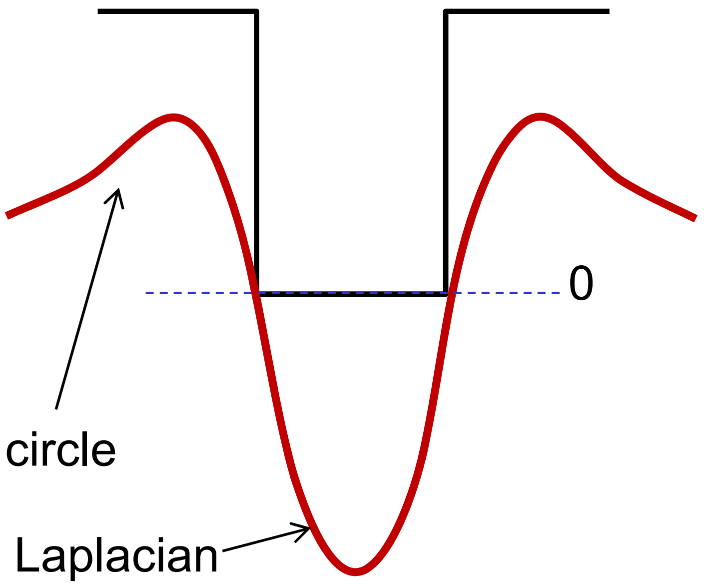
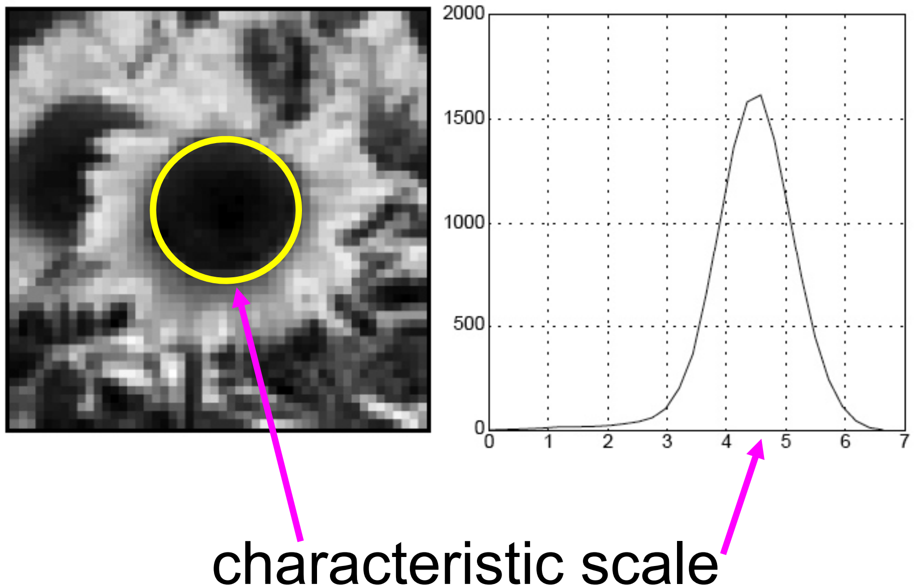


Image from http://www.cs.unc.edu/~lazebnik/spring11/lec08_blob.pdf

Characteristic scale of a blob

- The characteristic scale of a blob is the one that produces the maximum LoG response.

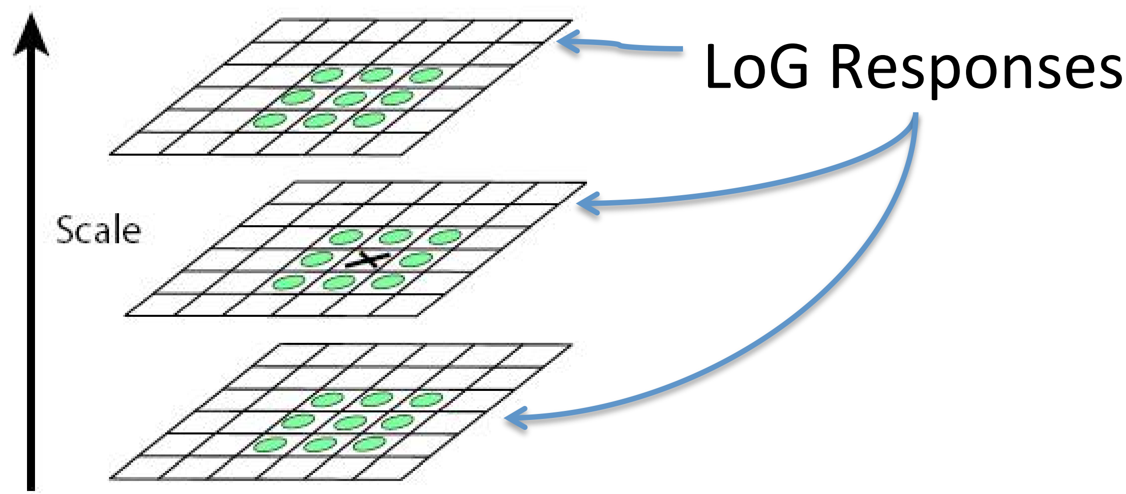


T. Lindeberg (1998). ["Feature detection with automatic scale selection."]() *International Journal of Computer Vision* **30** (2): pp 77--116.

Image from http://www.cs.unc.edu/~lazebnik/spring11/lec08_blob.pdf

Overview of Scale-space blob detector

1. Convolve the image at several scales with scale-normalized LoG.
2. Find the the maximal response in scale-space.



Why do you think corners or blobs are used more frequently than edges?

- Corners and blobs are repeatable
- Corners and blobs can be localized more easily