Computer Vision

5. Edge and Corner

I-Chen Lin

College of Computer Science

National Yang Ming Chiao Tung University

Objective

- Finding conspicuous low-level features (discontinuities) in images.
  - Edges
  - Corners

Textbook:
• David A. Forsyth and Jean Ponce, Computer Vision: A Modern Approach, Prentice Hall, New Jersey, (1st Ed. 2003, 2nd Ed. 2012).
Some contents are from the reference lecture notes:
• Prof. D. Lowe, Computer Vision, UBC, CA.
• D. Frolova, D. Simakov, Slides of “Matching with Invariant Features”.
• Prof. D.A. Forsyth, Computer Vision, UIUC.
• Prof. J. Rehg, Computer Vision, Georgia Inst. of Tech.
• Prof. T. Darrell, Computer Vision and Applications, MIT.

What causes an edge?

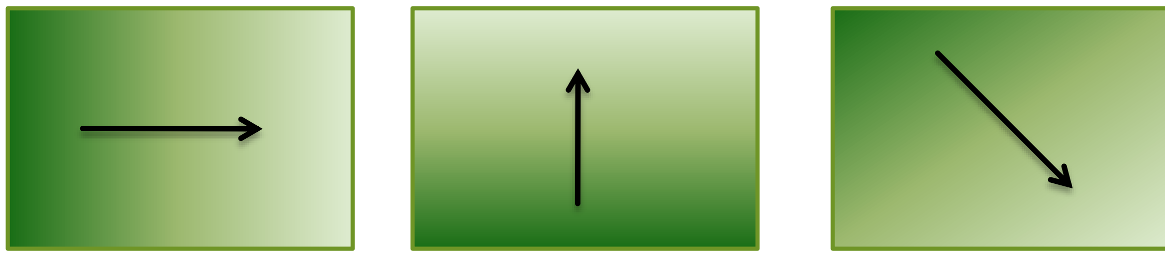
- Depth discontinuity
- Surface orientation discontinuity
- Reflectance discontinuity (i.e., change in surface material properties)
- Illumination discontinuity (e.g., shadow)

Gradient

- The gradient of an image:

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

  - E.g.



$$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right] \qquad \nabla f = \left[0, \frac{\partial f}{\partial y}\right] \qquad \nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

- The gradient direction:

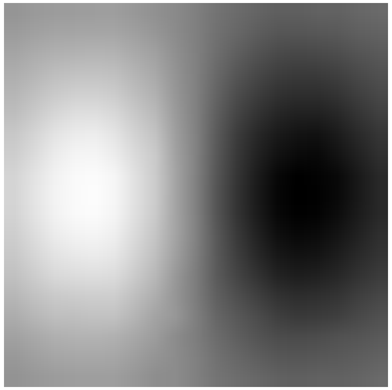
$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} \middle/ \frac{\partial f}{\partial x}\right)$$

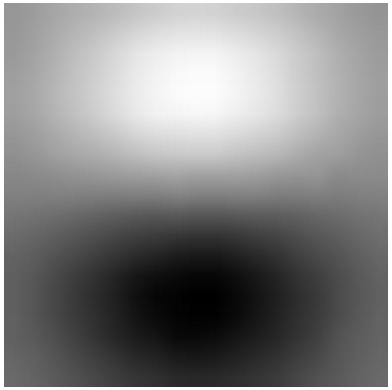
- The gradient magnitude:

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

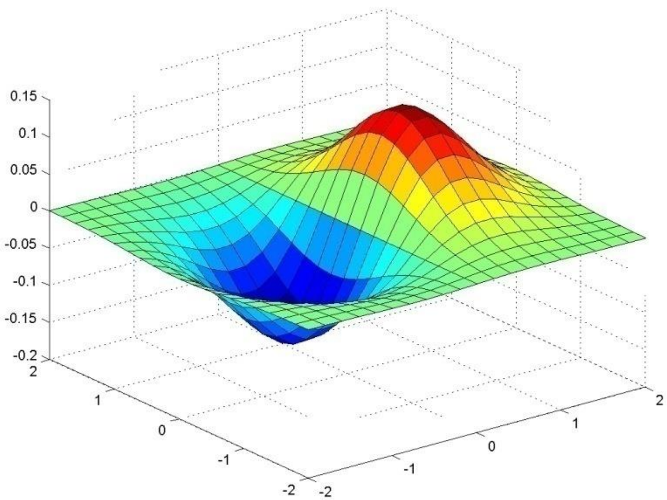
Edge and differentiation

- **Edge:** a location with high gradient (derivative).
- Need smoothing to reduce noise prior to taking derivative.
- Two derivatives, in x and y direction for an image.
- We can use derivative of Gaussian filters
  - because differentiation is convolution, and convolution is associative: D * (G * I) = (D * G) * I

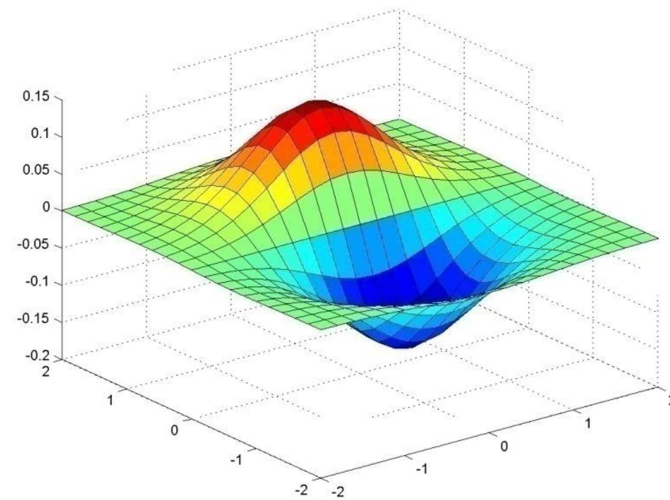




Derivative of Gaussian



$$\frac{\partial}{\partial x}G_\sigma$$

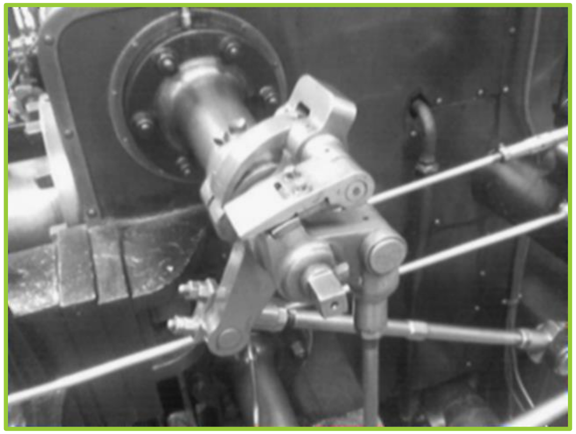


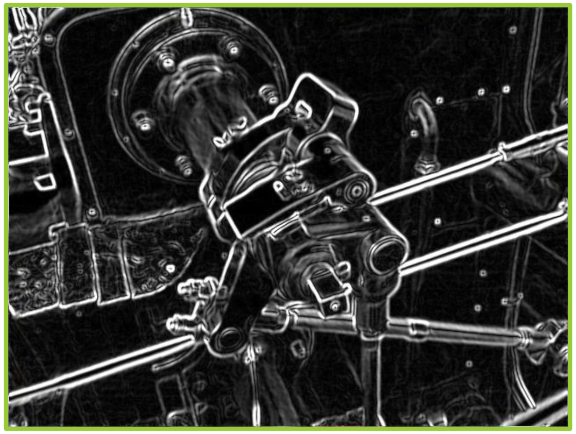
$$\frac{\partial}{\partial y}G_\sigma$$

Slide credit: Christopher Rasmussen, from lecture notes of Prof. D. Lowe, Computer Vision, UBC, CA

Sobel Edge Detection

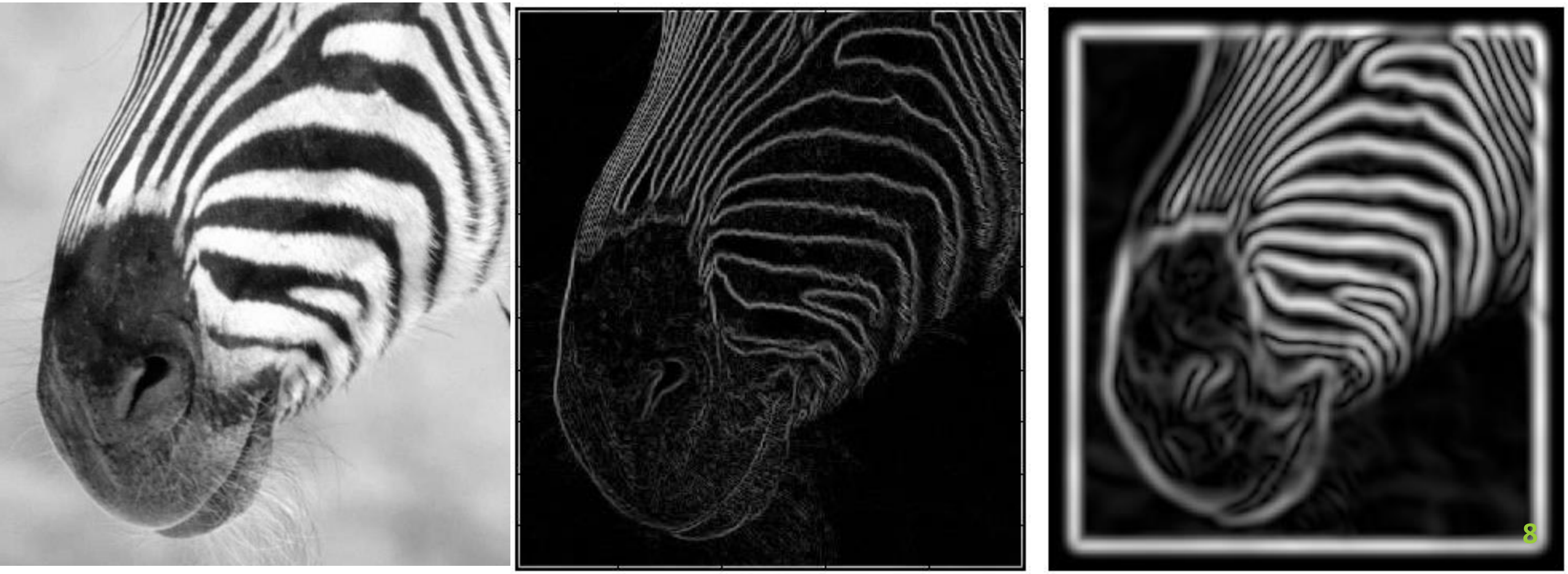
$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * \mathbf{A} \quad \text{and} \quad \mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathbf{A}$$





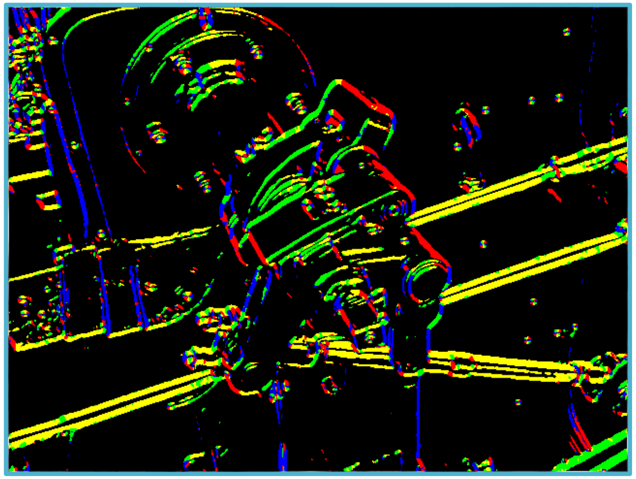
Gradient magnitude and smoothing

- Increase smoothing:
  - Eliminates noise edges.
  - Makes edges smoother and thicker.
  - Removes fine detail

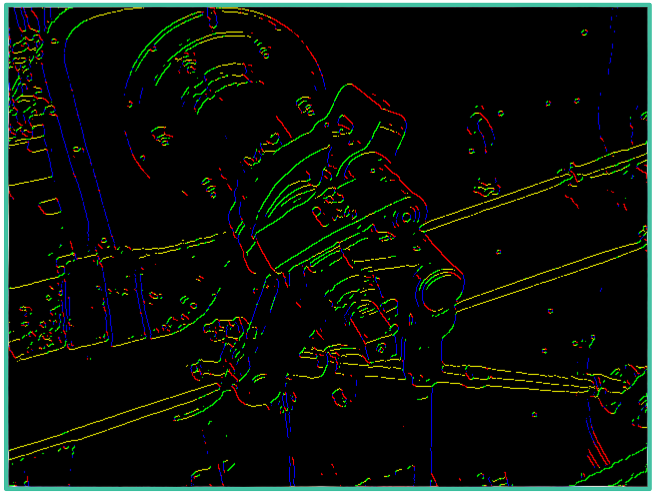


Canny edge detection

- Noise removal (Gaussian filtering)
- Gradients of the image
- Non-maximum suppression
  - Check whether a pixel is a local maximum along gradient direction



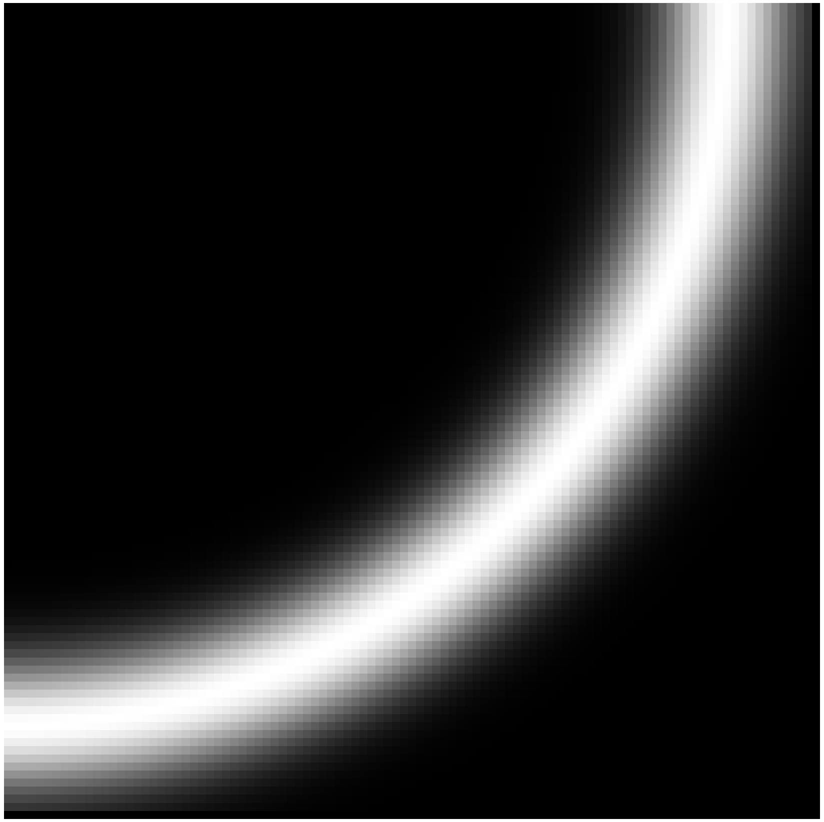
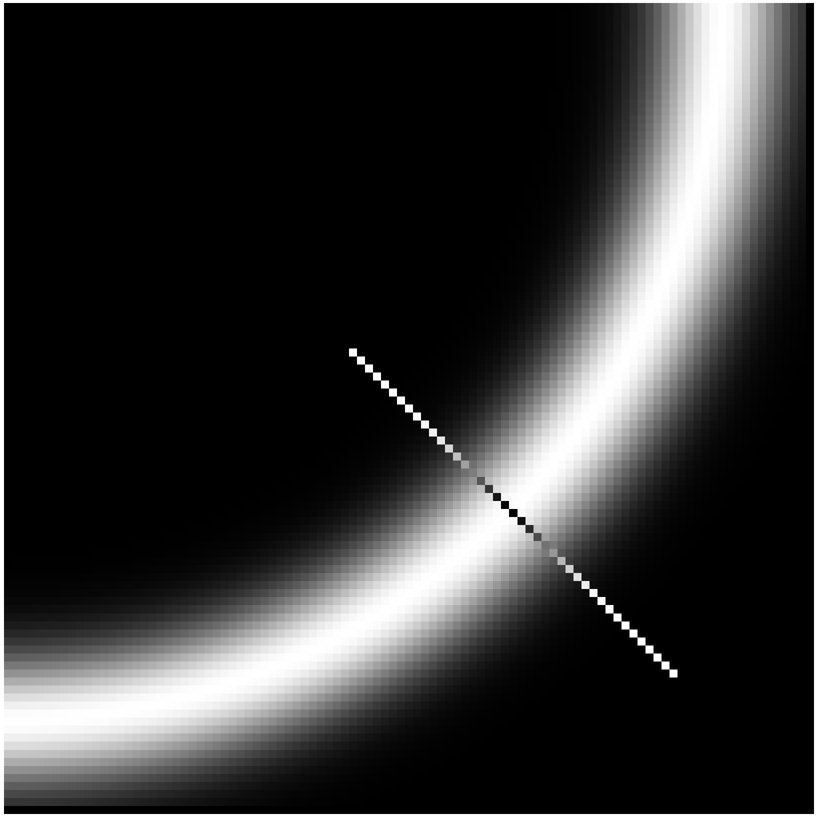
Yellow for 0 degrees; green for 45 degrees; blue for 90 degrees ; red for 135 degrees



After non-maximum suppression

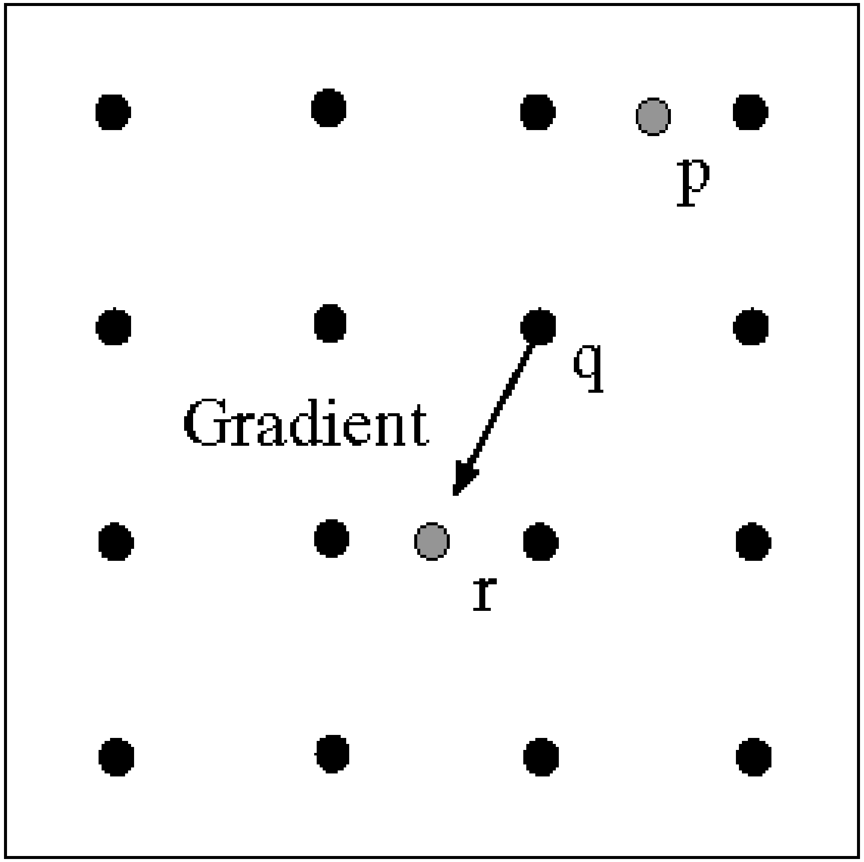
Non-maximum suppression

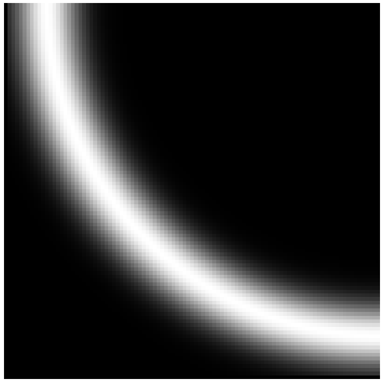
- Select the single maximum point across the width of an edge.

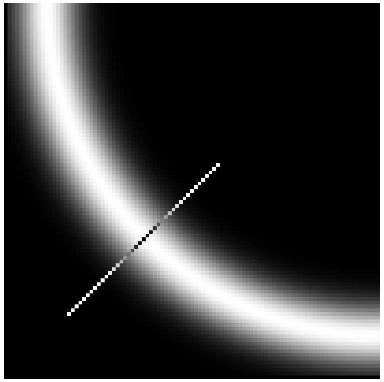



Non-maximum suppression

- At *q*, the value must be larger than values interpolated at *p* or *r*.







Examples non-maximum suppression

Original image

Gradient magnitude

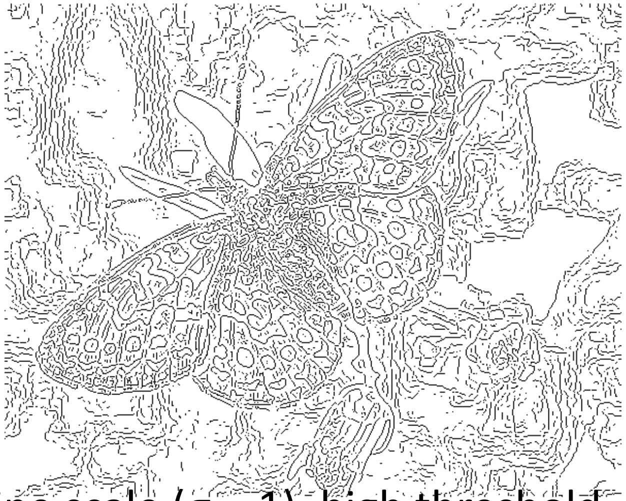
courtesy of G. Loy

Non-maxima suppressed

Slide credit: Christopher Rasmussen, from lecture notes of Prof. D. Lowe, Computer Vision, UBC, CA

Examples non-maximum suppression

coarse scale, (σ = 4), high threshold

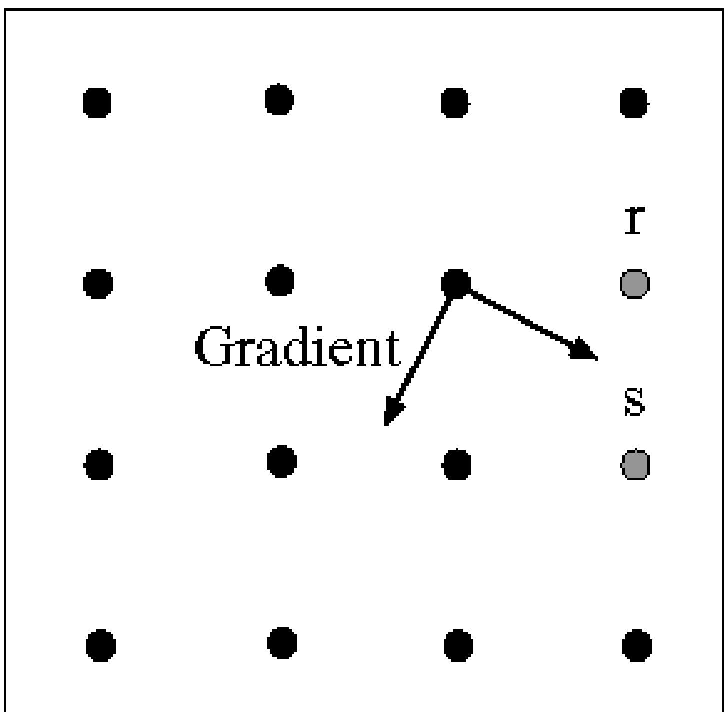


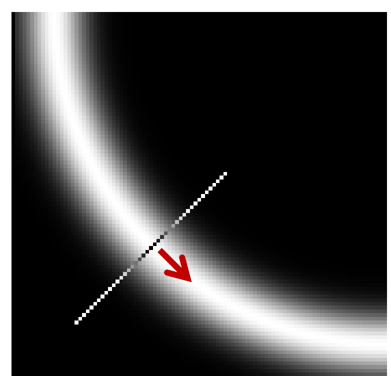
fine scale (σ = 1), high threshold

coarse scale (σ = 4), low threshold

Linking to the next edge point

- Assume the marked point is an edge point.
- Take the normal to the gradient at that point and use this to predict continuation points (either *r* or *s*).





Edge hysteresis

- Hysteresis: A lag or momentum factor.
- Idea: Maintain two thresholds k_{high} and k_{low}
- Use k_{high} to find strong edges to start edge chain
- Use $_{klow}$ to find weak edges which continue edge chain
- Typical ratio of thresholds is roughly

  $k_{high} / k_{low} = 2$

Steps of Canny edge detection

- Apply derivative of Gaussian
- Non-maximum suppression
  - Thin multi-pixel wide “ridges” down to single pixel width
- Linking and thresholding
  - Low, high edge-strength thresholds
  - Accept all edges over low threshold that are connected to edge over high threshold

Example: Canny edge detection

Original image

gap is gone

Strong + connected weak edges

Strong edges only

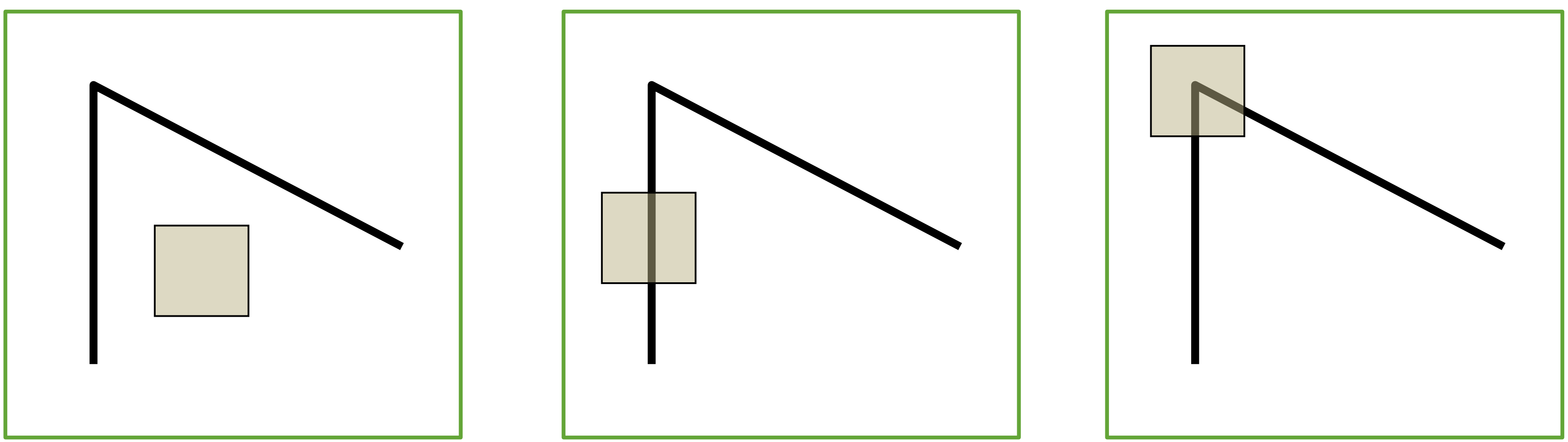
Weak edges

courtesy of G. Loy

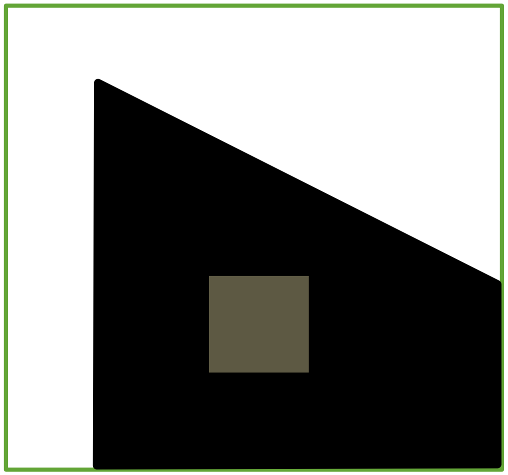
Figures from lecture notes of Prof. D. Lowe, Computer Vision, UBC, CA

Conspicuous location

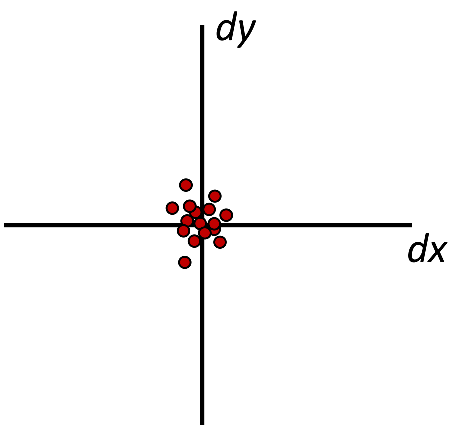
- First of all, we would like to find “unique” or “conspicuous” positions.
- For a small searching window, which one is an “unique” place?

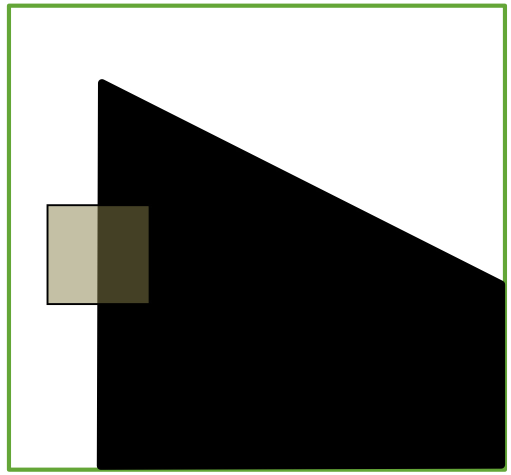


Conspicuous location

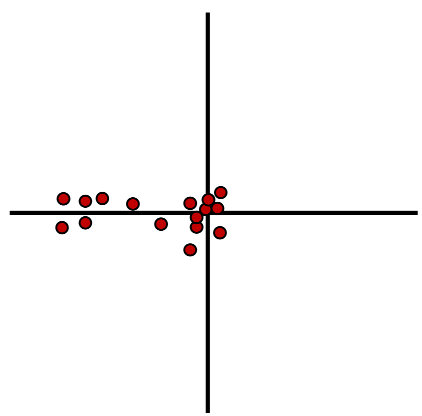


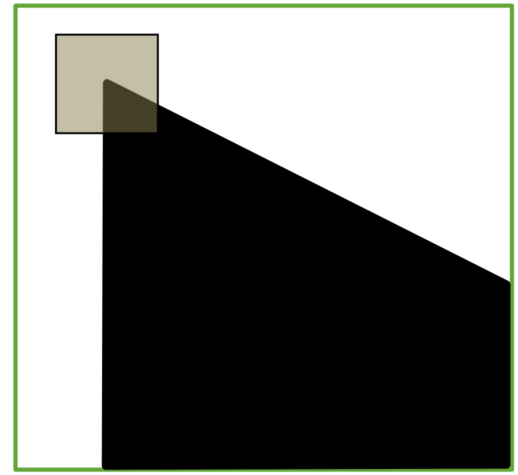
“flat” region:
no change in all directions



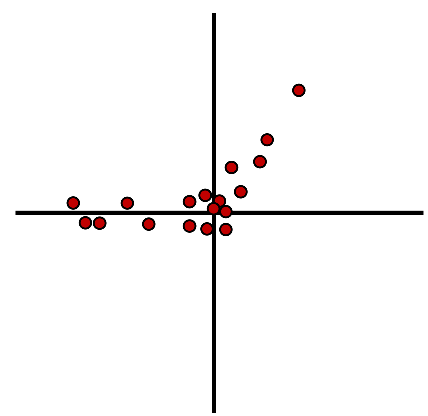


“edge”:
no change along the edge direction





“corner”:
significant change in all directions



Finding corners

- Edge detectors perform poorly at corners.
- Corners provide repeatable points for matching, so are worth detecting.

- Idea:
  - Exactly at a corner, gradient is ill defined.
  - However, in the region around a corner, gradient has two or more different values.

Corner detection (Harris)

- Consider the matrix for a small square around (*x*,*y*)

$$I_x = \frac{\partial I}{\partial x} \qquad I_y = \frac{\partial I}{\partial y}$$

$$A = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$$

$$A = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \approx \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

- The simplest case
  - If $\lambda_1 \fallingdotseq 0$ and $\lambda_2 \fallingdotseq 0$ then there are no features of interest at this pixel (*x*,*y*).
  - If $\lambda_1 \fallingdotseq 0$ and λ_2 is some large positive values, then an edge is found.
  - If λ_1 and λ_2 are both large, distinct positive values, then a corner is found.

Corner detection (Harris)

- More general cases:

$$A = Rot^{-1}\begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} Rot$$

where Rot, Rot^{-1} can be regarded as rotation matrices.

  - Remind “Eigenvalues” or “Singular Value Decomposition (SVD)”.

- Process steps
  - Apply Gaussian filter.
  - Evaluate magnitudes of the gradients.
  - Construct A.
  - Find λ_1 and λ_2 by evaluation of eigen values or SVD.
  - If they are both big, we have a corner.

Harris detector: mathematics

Explicitly evaluate the eigen values

$$H = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad \lambda_{\pm} = \frac{1}{2}\left((a+d) \pm \sqrt{4bc + (a-d)^2}\right)$$

Or measure of corner response *R*:

$$R = \det M - k\left(\operatorname{trace} M\right)^2$$

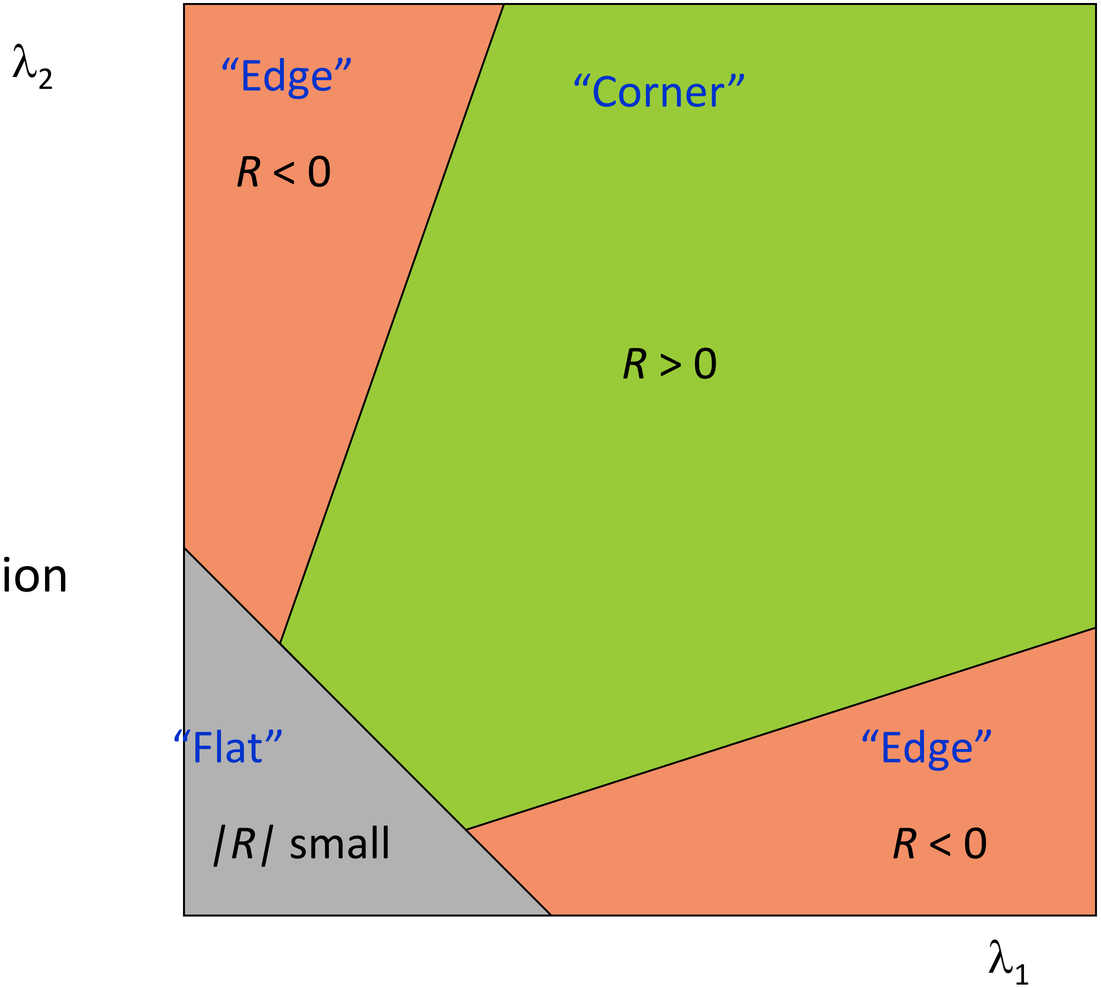
$$\det M = \lambda_1 \lambda_2$$

$$\operatorname{trace} M = \lambda_1 + \lambda_2$$

(*k* – empirical constant, *k* = 0.04-0.06)

Harris detector: mathematics

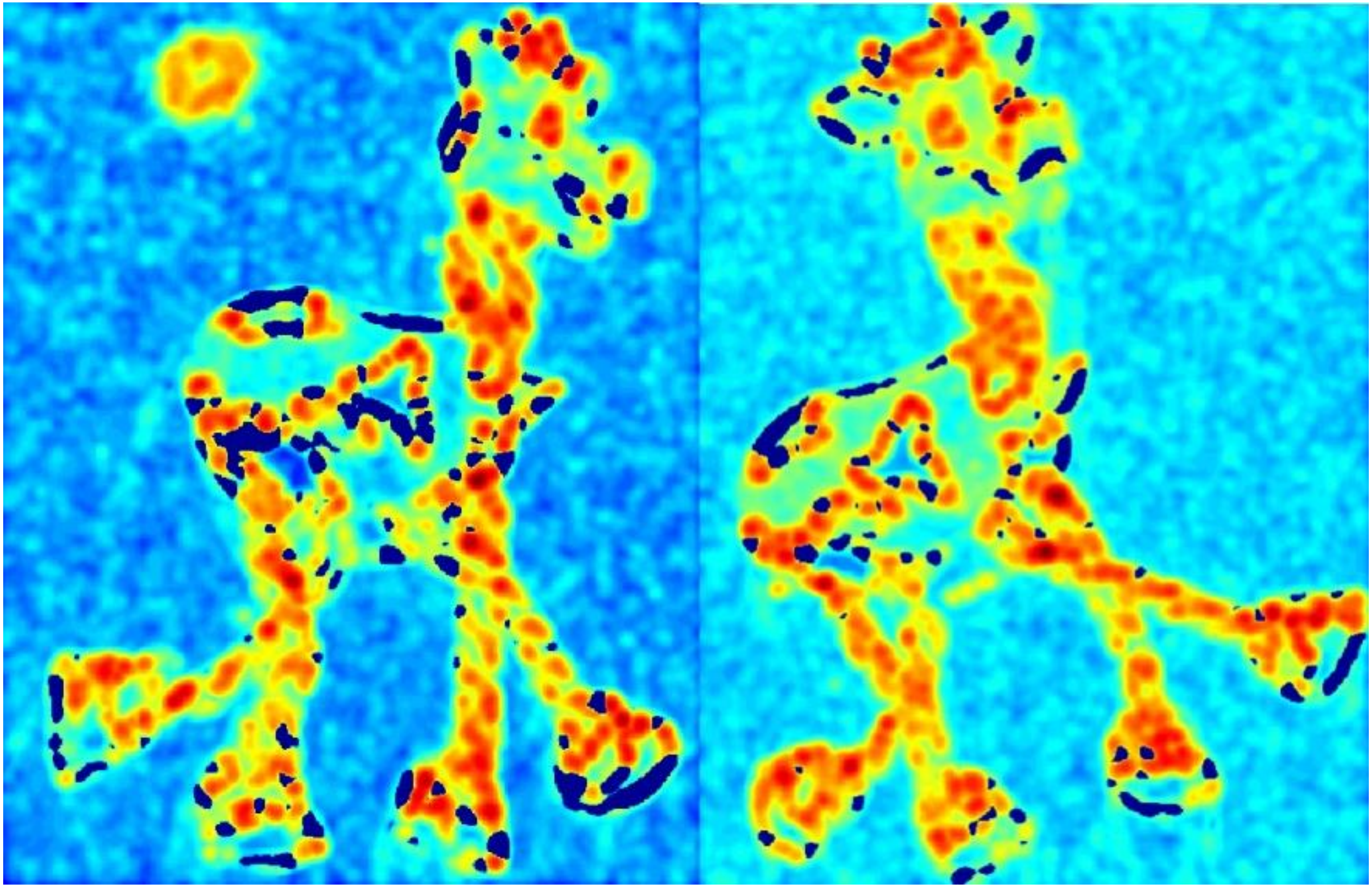
- R depends only on eigenvalues of M
- R is large for a corner
- R is negative with large magnitude for an edge
- $|R|$ is small for a flat region



Harris detector: workflow

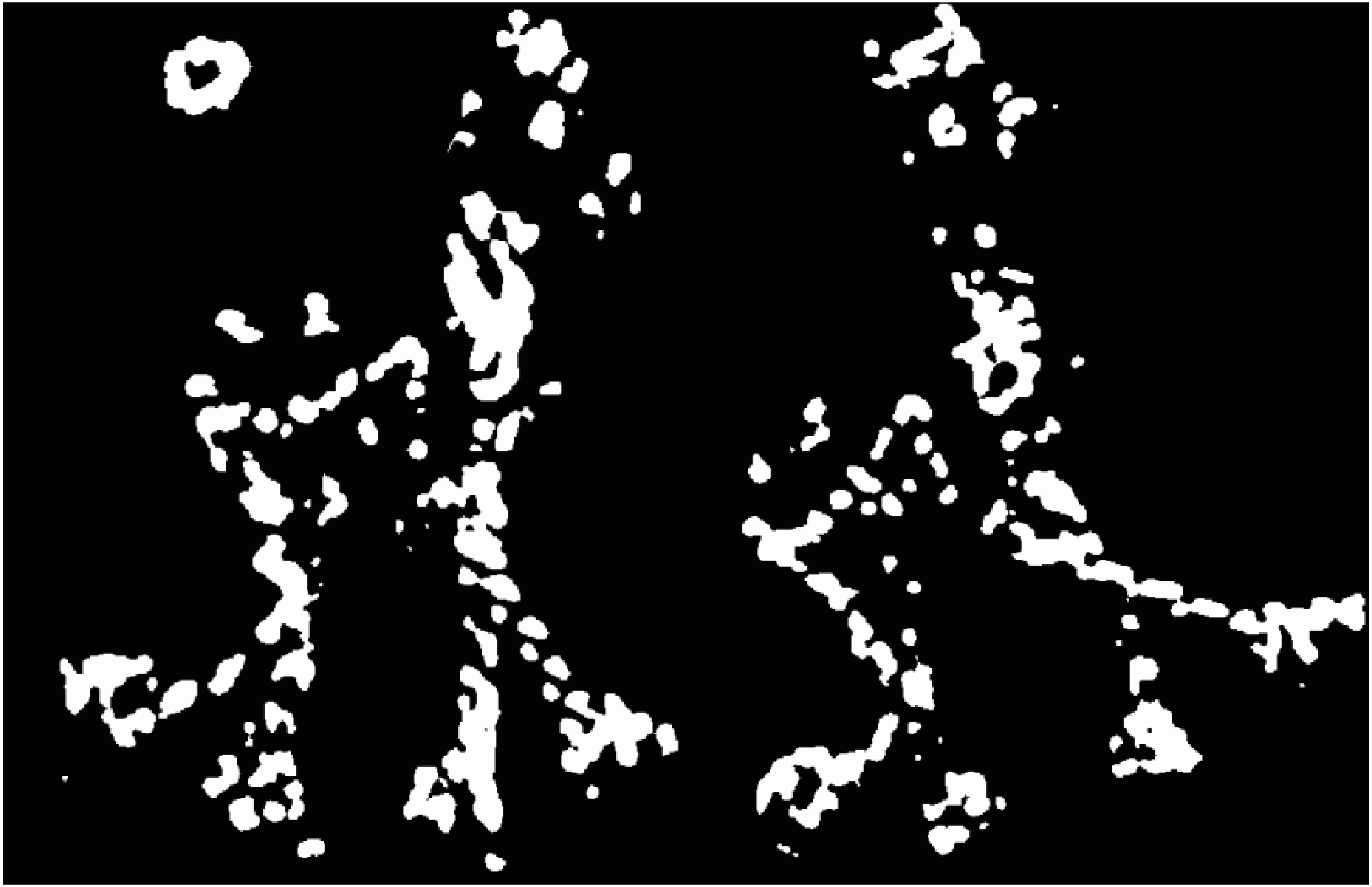
Harris detector: workflow

Compute corner response



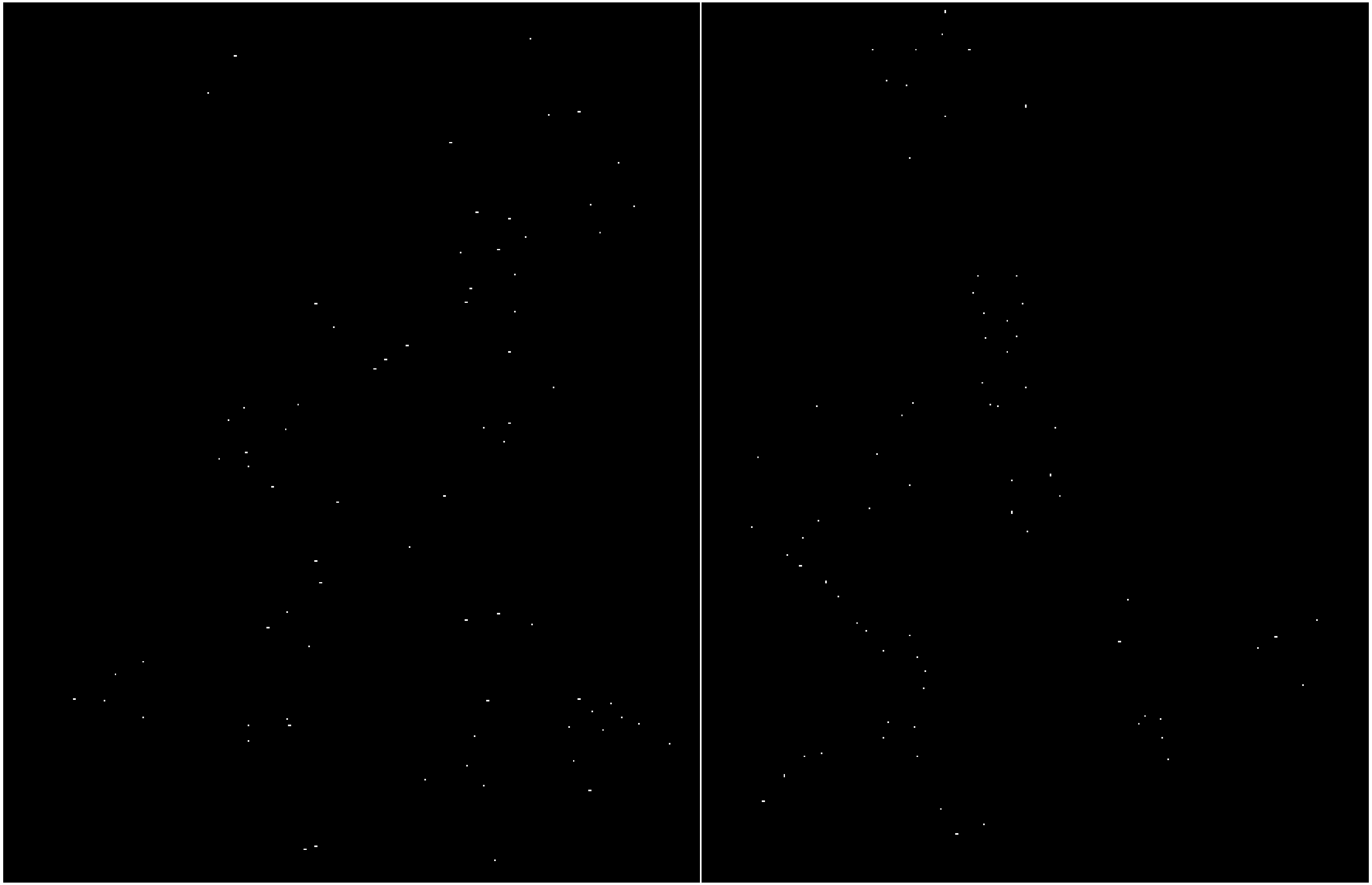
Harris detector: workflow

Find points with large corner response (>threshold)



Harris detector: workflow

Take only the points of local maxima

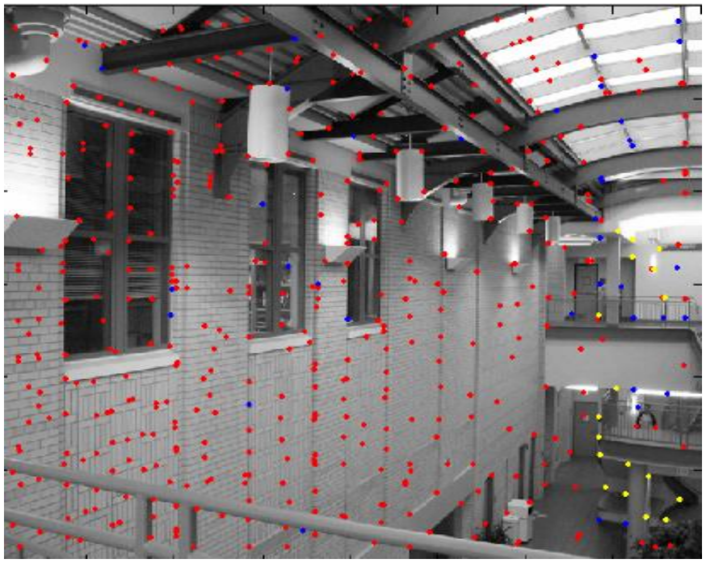


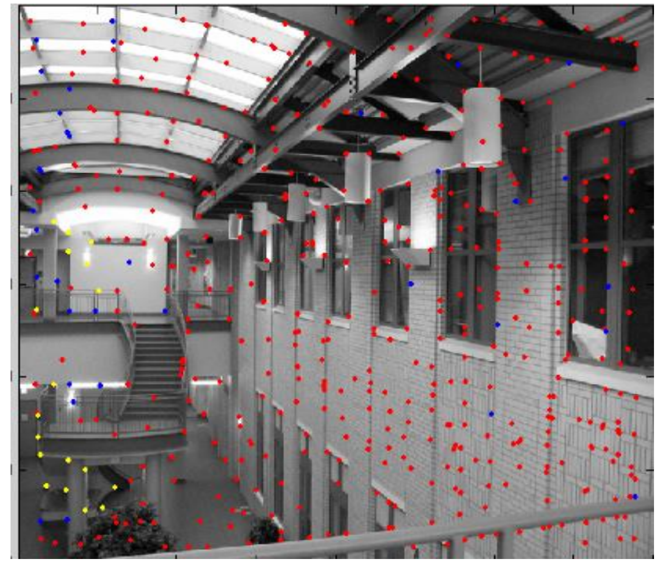
Harris detector: workflow

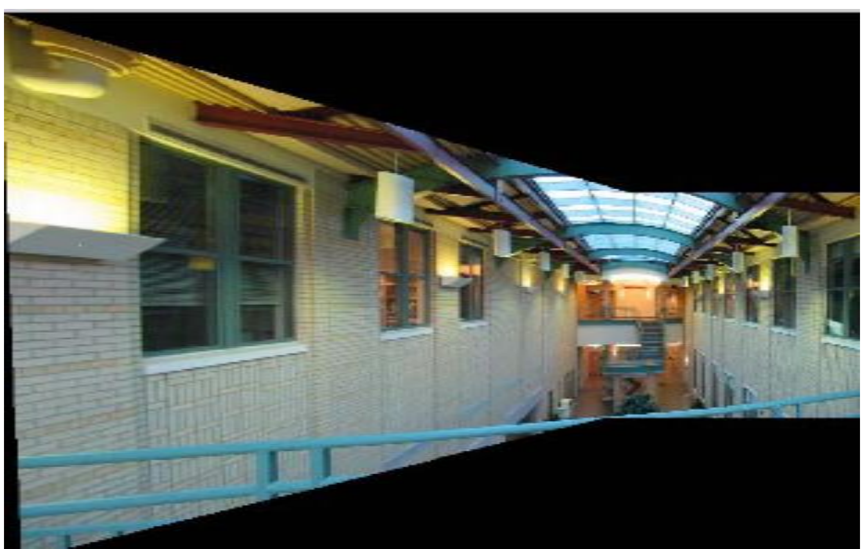
Harris corners

- Originally developed as features for motion tracking
- Greatly reduces amount of computation compared to tracking every pixel
- Translation and rotation invariant (but not scale invariant)

Matching with corner and features







Figures from Alexei Efros, Computational Photography