Edge Detection and Corner Detection

Introduction to Computer Vision I

CSE 152A

Lecture 6

Announcements

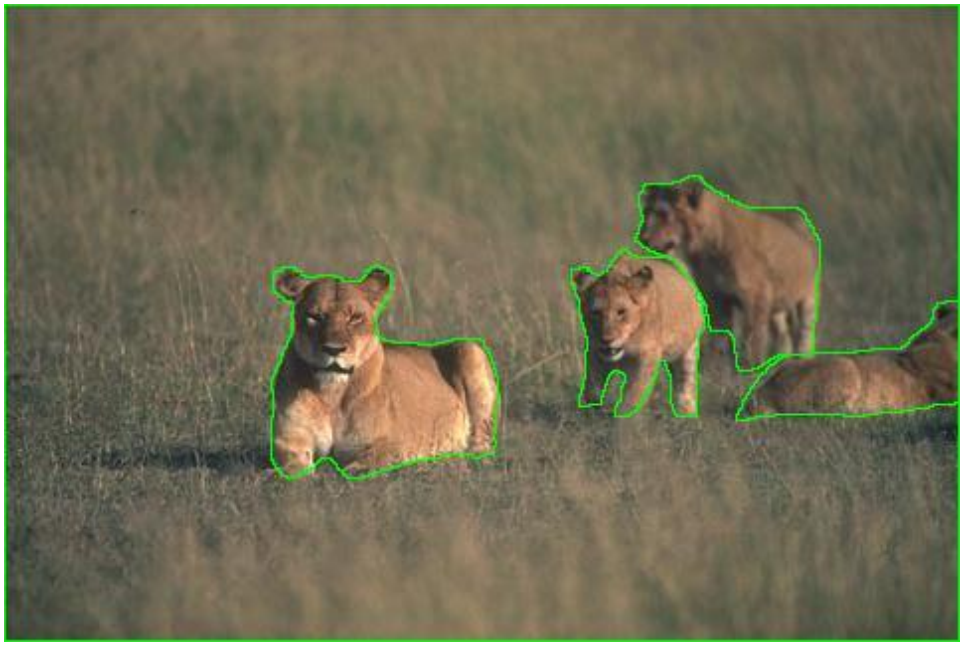
- Assignment 1 is due Oct 21, 11:59 PM
- Assignment 2 will be released Oct 21
  - Due Nov 4, 11:59 PM

Image Segmentation and Edges

- Image Segmentation is the process of dividing an image into connected regions such that pixels within a region share certain characteristics (color, texture , brightness, etc.)
- Boundaries or edges divide segmented regions.

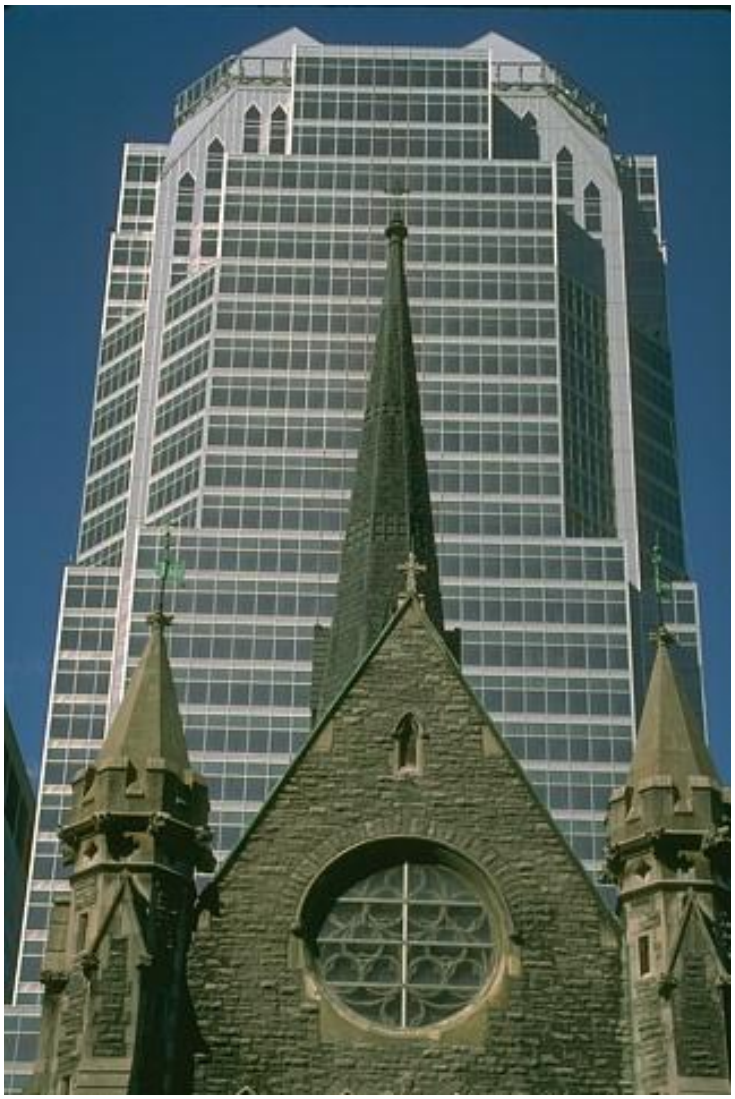


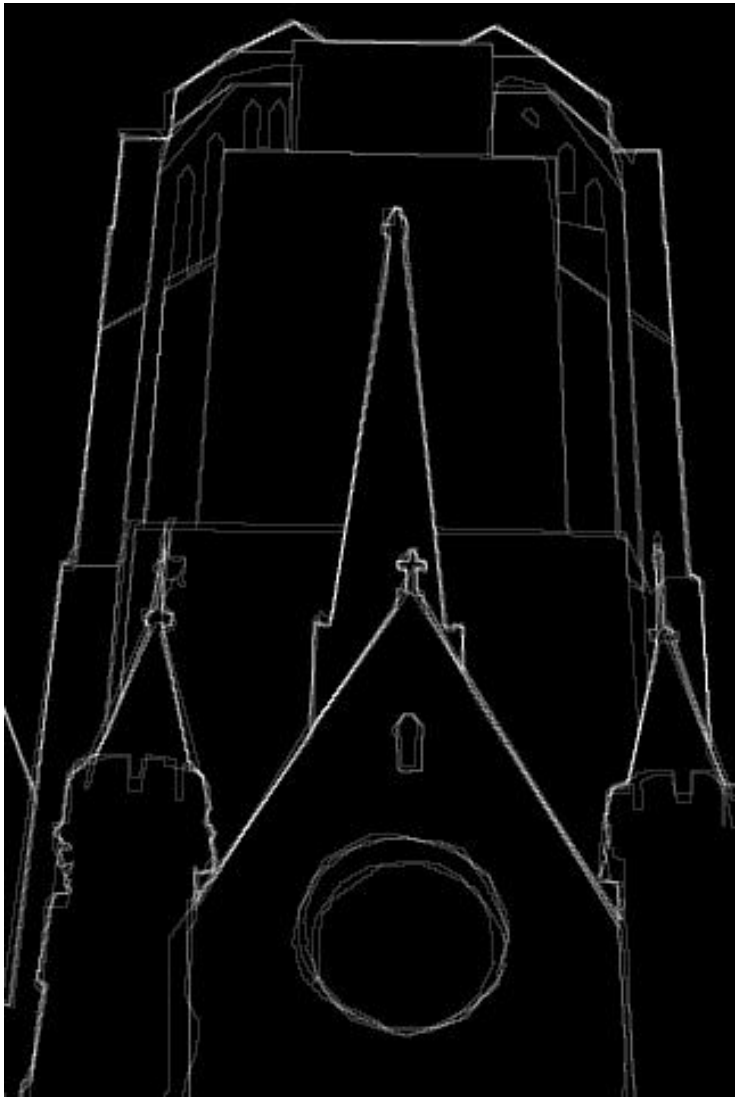
[From Berkeley Segmentation Dataset]



13 Regions

Image Segmentation



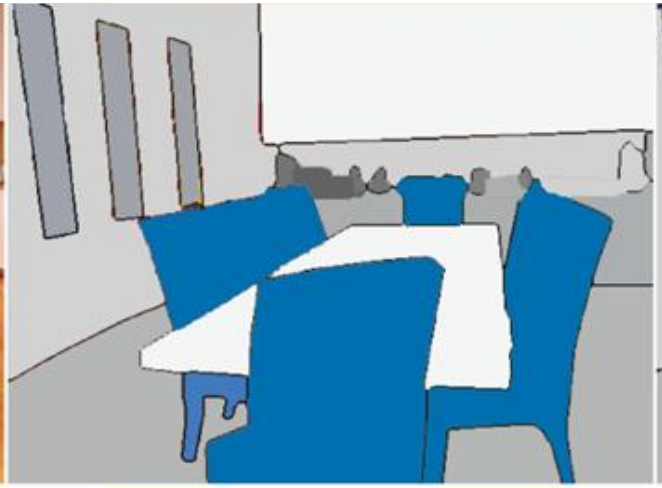


[From Berkeley Segmentation Dataset]

Related Topics: Semantic and Instance Segmentation



Input Image



Semantic Segmentation



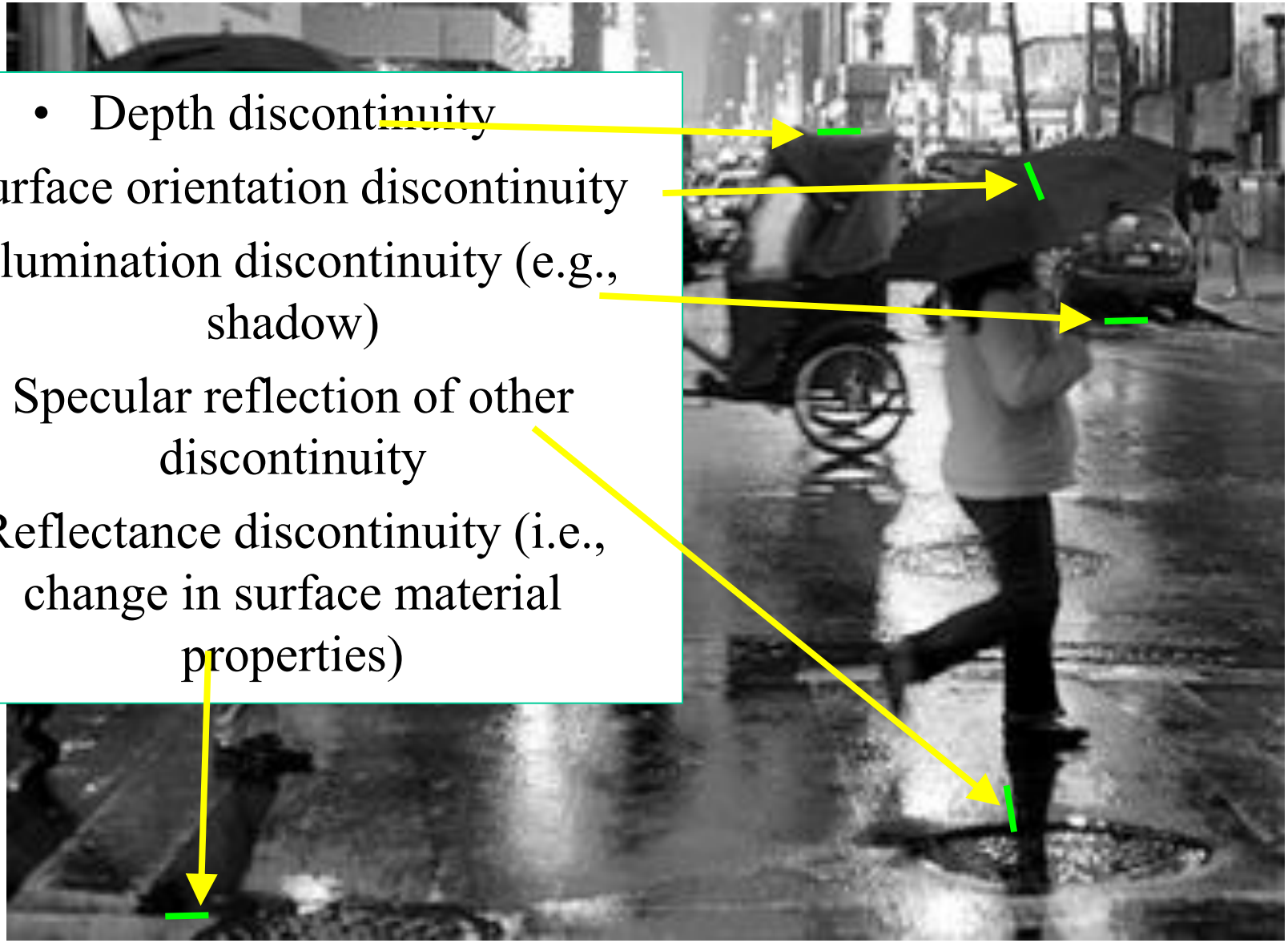
Instance Segmentation

Edges in Natural Images



Source: Photografr.com

What Causes an Edge?

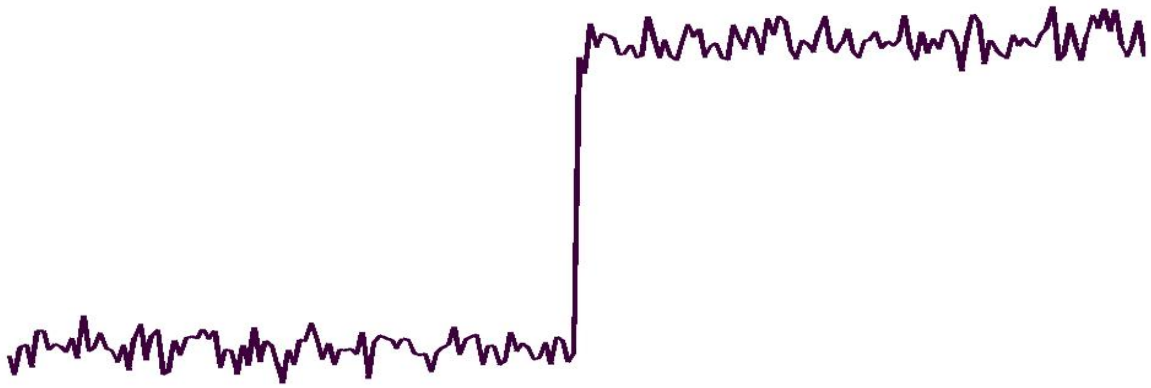


- Depth discontinuity
- Surface orientation discontinuity
- Illumination discontinuity (e.g., shadow)
- Specular reflection of other discontinuity
- Reflectance discontinuity (i.e., change in surface material properties)

Source: Photografr.com

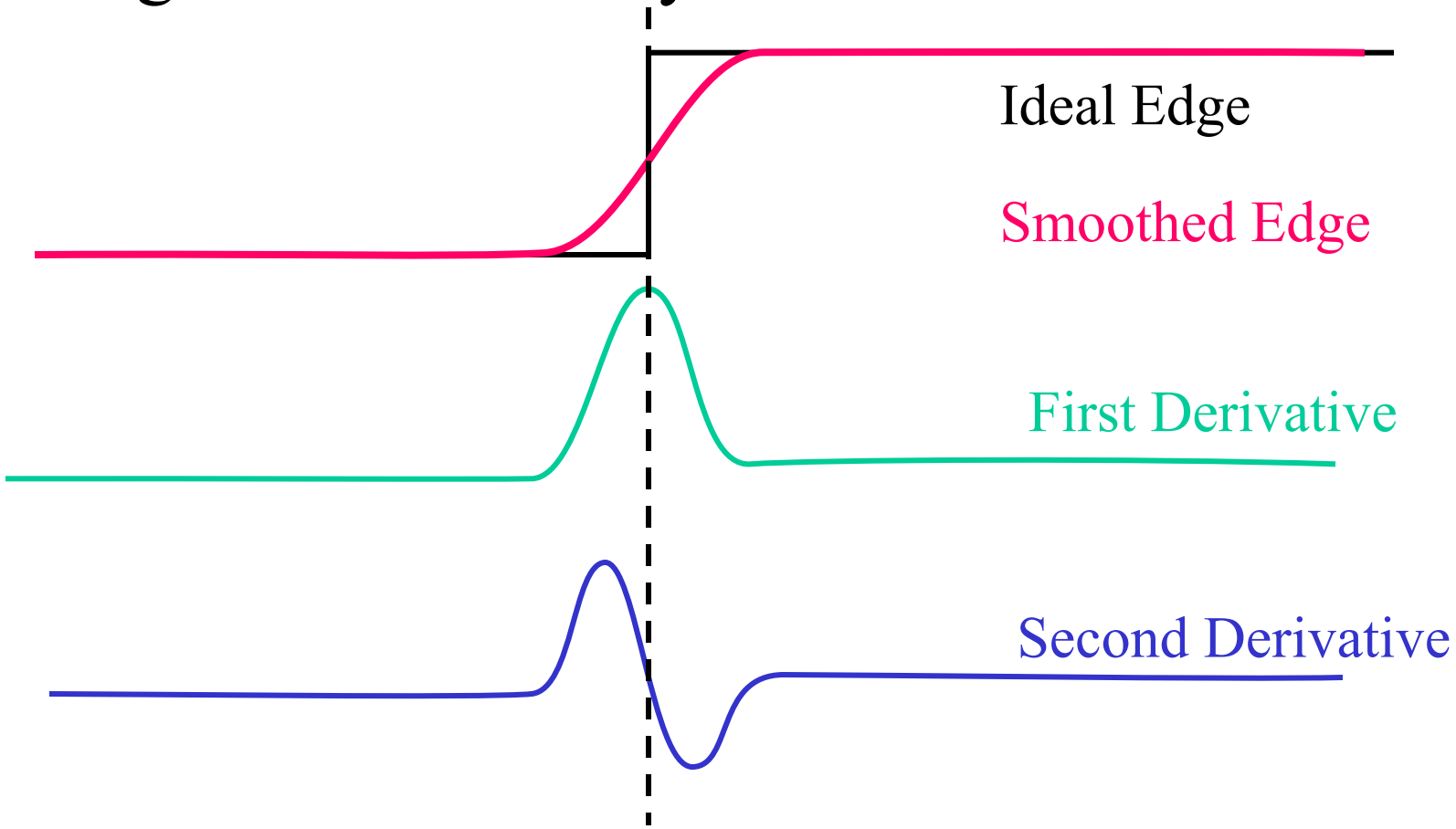
Noisy 1D Step Edge

- Derivative is high everywhere.
- Must smooth before taking gradient.



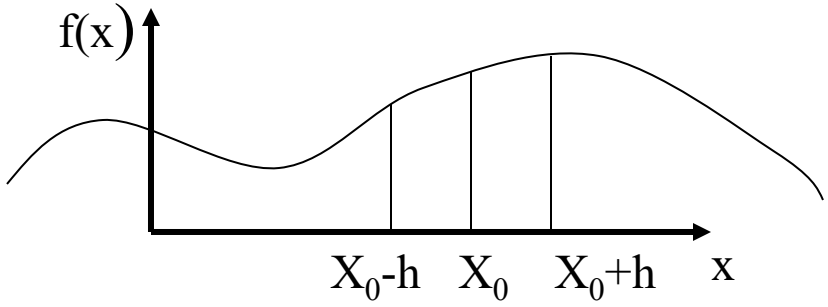
Edge is Where Change Occurs: 1-D

- Change is measured by derivative in 1D



- Biggest change, first derivative has maximum magnitude
- Or second derivative is zero

Numerical Derivatives of Sampled Signal



Take Taylor series expansion of f(x) about x_0

$$f(x) = f(x_0)+f'(x_0)(x-x_0) + \tfrac{1}{2} f''(x_0)(x-x_0)^2 + \cdots$$

Consider samples taken at increments of h and first two terms of the expansion, we have

$$f(x_0+h) = f(x_0)+f'(x_0)h+ \tfrac{1}{2} f''(x_0)h^2$$

$$f(x_0-h) = f(x_0)-f'(x_0)h+ \tfrac{1}{2} f''(x_0)h^2$$

Subtracting and adding $f(x_0+h)$ and $f(x_0-h)$ respectively yields

$$f'(x_0) = \frac{f(x_0+h)-f(x_0-h)}{2h}$$

$$f''(x_0) = \frac{f(x_0+h)-2f(x_0)+f(x_0-h)}{h^2}$$

Correlate with
First Derivative: [-1/2h 0 1/2h]
Second Derivative: [$1/h^2$ $-2/h^2$ $1/h^2$]

Numerical Derivatives

Correlate with
First Derivative: [-1/2h 0 1/2h]
Second Derivative: [$1/h^2$ $-2/h^2$ $1/h^2$]

- With images, units of h is pixels, so h=1
  - First derivative: [-1/2 0 1/2]
  - Second derivative: [1 -2 1]
- When computing derivatives in the x and y directions, use these kernels with correlation:

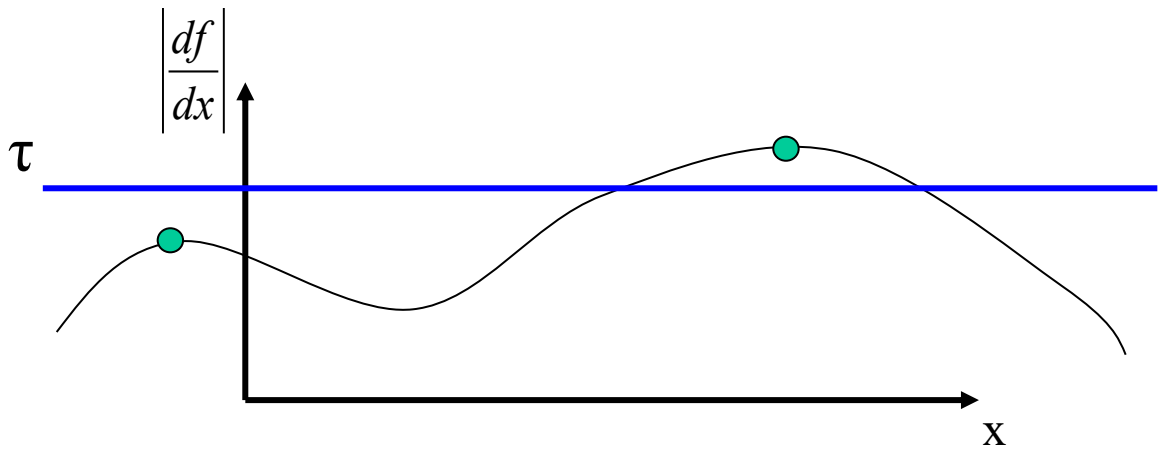
$$\frac{d}{dx} = [-1/2 \quad 0 \quad 1/2] \qquad \frac{d}{dy} = \begin{bmatrix} -1/2 \\ 0 \\ 1/2 \end{bmatrix}$$

Implementing 1D Edge Detection

1. Filter out noise: convolve with Gaussian
2. Take a derivative: convolve with [1/2 0 -1/2]

   We can combine steps 1 and 2
3. Find the peaks of |df/dx|

   Two issues:
   - Should be a local maximum of |df/dx|
   - Should be greater than a threshold: $|df/dx| > \tau$



2D Edge Detection

1. Filter out noise
   - Use a 2D Gaussian Filter. $J = I \star G$
2. Take derivatives
   - Compute the magnitude of the gradient:

$$\nabla J = \left(\frac{\partial J}{\partial x}, \frac{\partial J}{\partial y} \right)^\top = (J_x, J_y)^\top \quad \text{gradient vector}$$

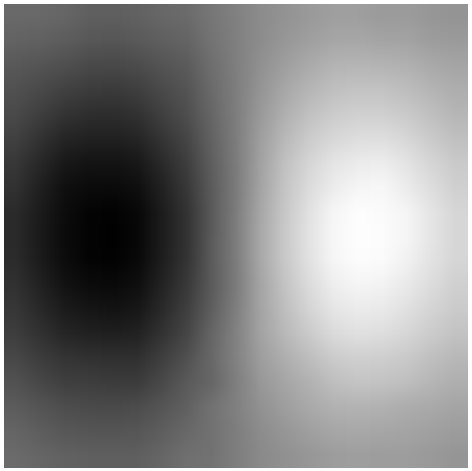
$$\|\nabla J\| = \sqrt{J_x^2 + J_y^2} \quad \text{magnitude of gradient}$$

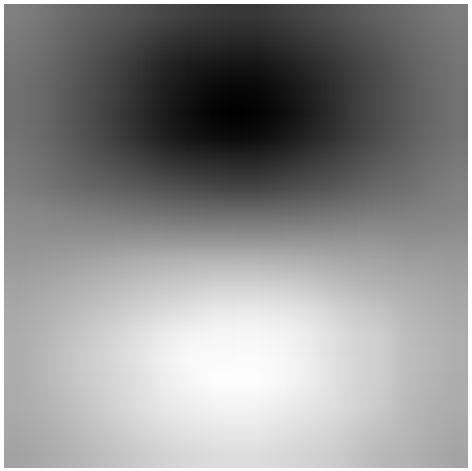
$$\alpha(x, y) = \tan^{-1}\left(\frac{J_y}{J_x} \right) \quad \text{direction of gradient}$$

Use atan2

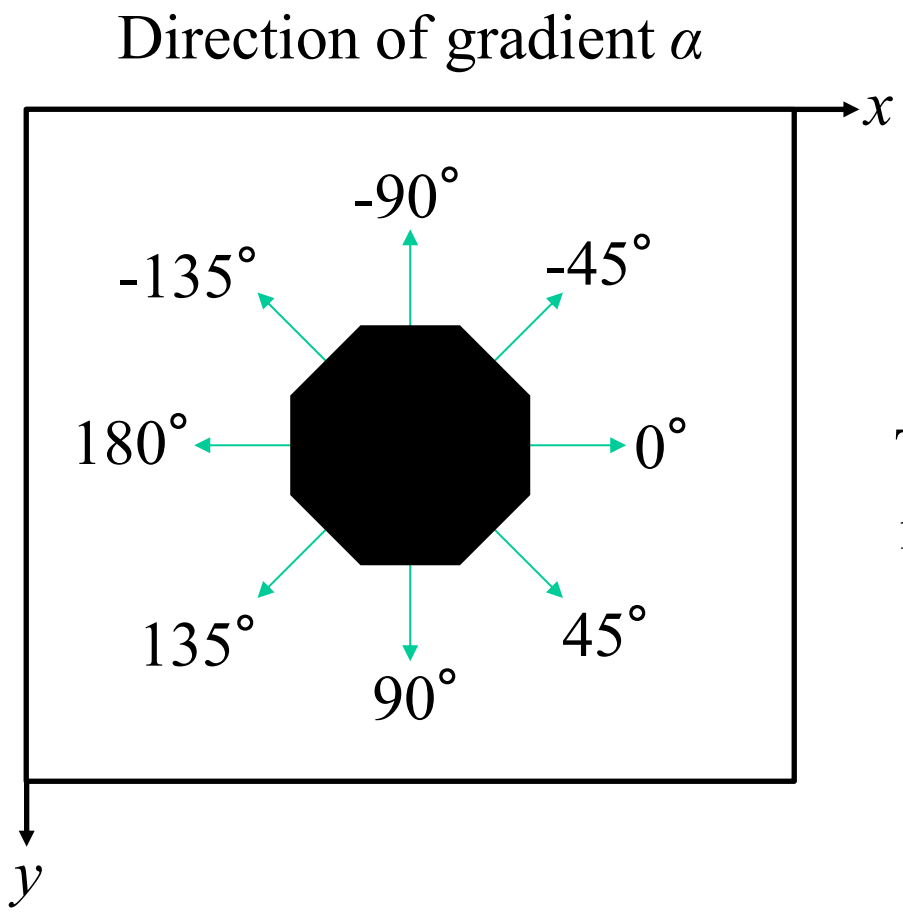
Smoothing and Differentiation

- Need two derivatives, in x and y direction.
- Filter with Gaussian and then compute Gradient, OR
- Use a derivative of Gaussian filter
    - because differentiation using convolution, and convolution is associative (shape full convolution is required)





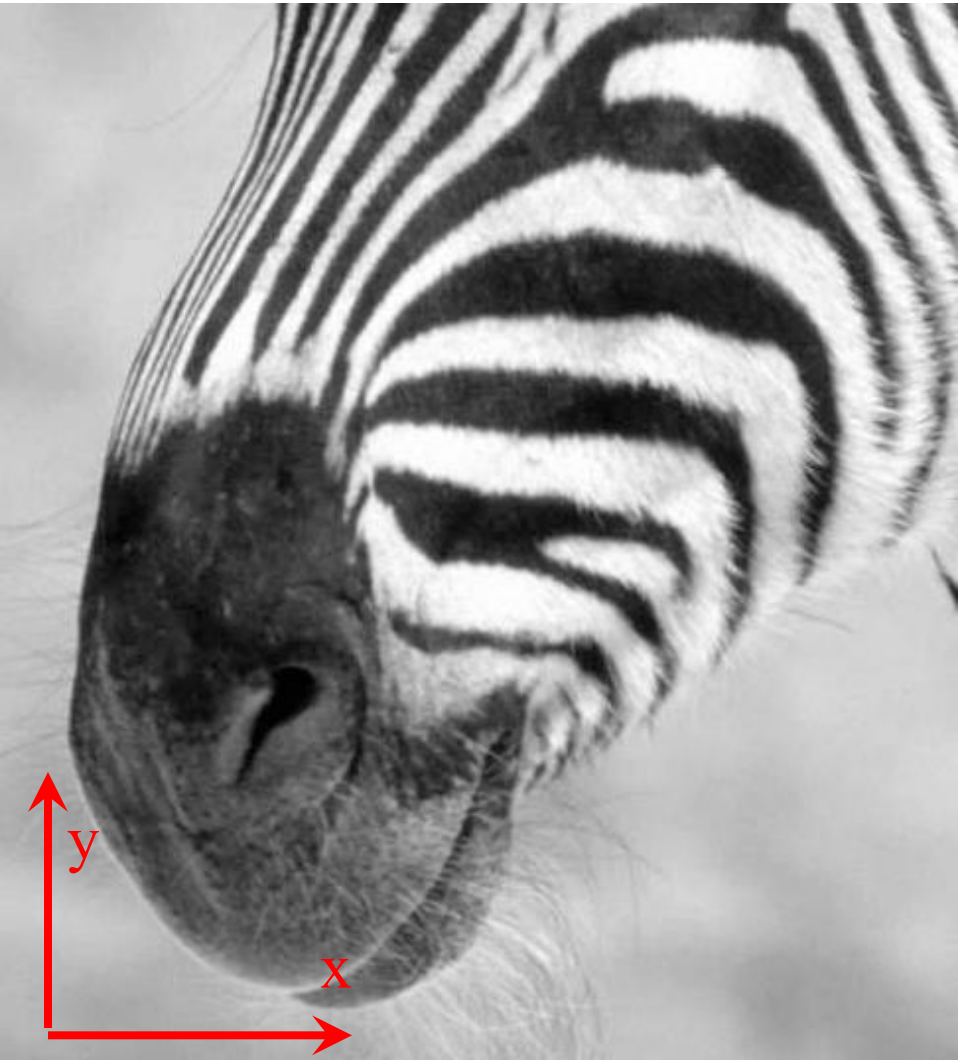
Gradient vector direction

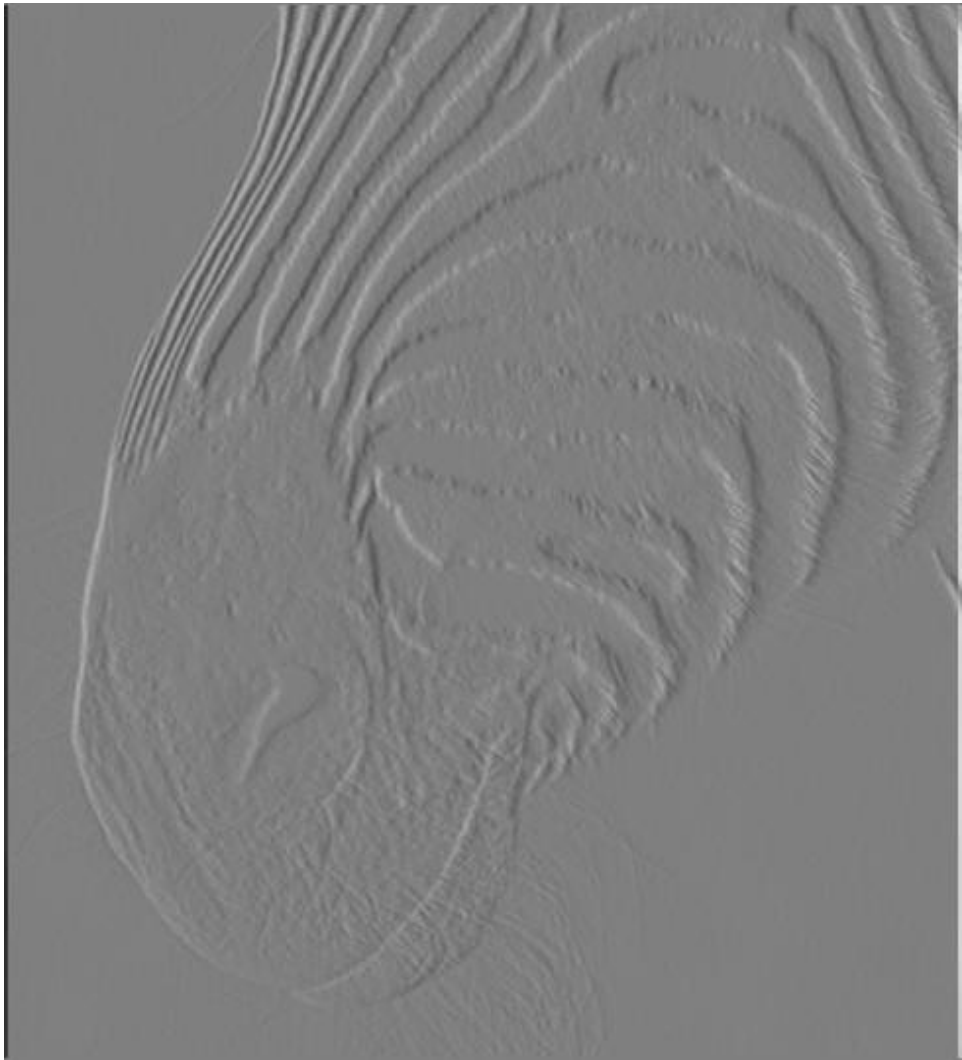


The gradient vector is one of the edge normal vectors. The other normal edge vector is in the opposite direction of the gradient vector.

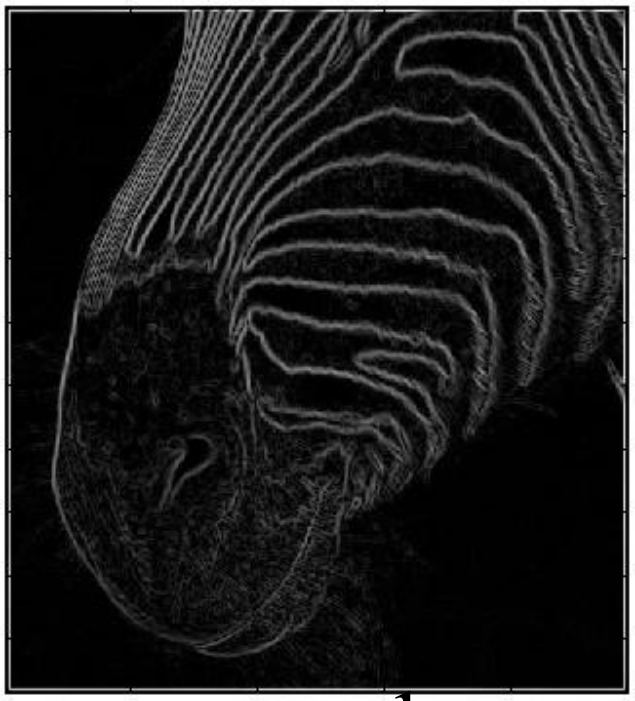
Finding derivatives

Is this dI/dx or dI/dy?









$\sigma = 1$



$\sigma = 2$

There are three major issues:

1. The gradient magnitude at different scales is different; which scale should we choose?
2. The gradient magnitude is large along a thick trail; how do we identify the significant points?
3. How do we link the relevant points up into curves?

There is ALWAYS a tradeoff between smoothing and good edge localization!

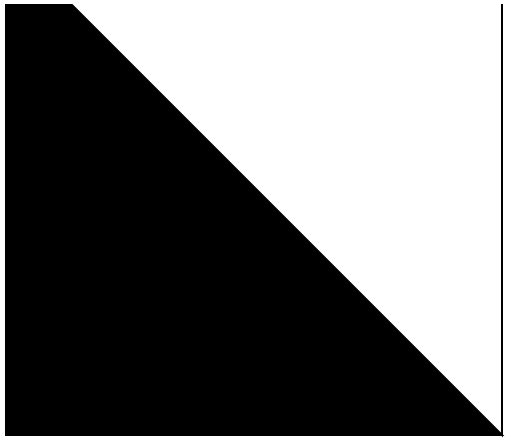
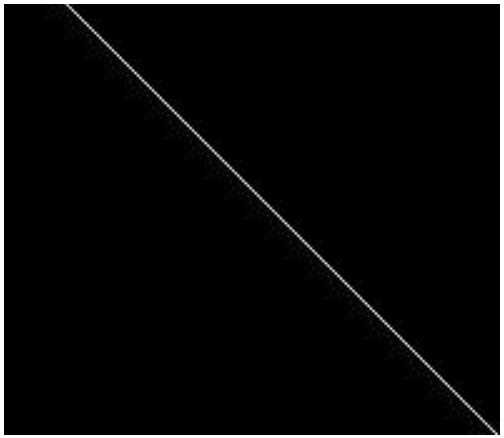


Image with Edge (No Noise)



Edge Location

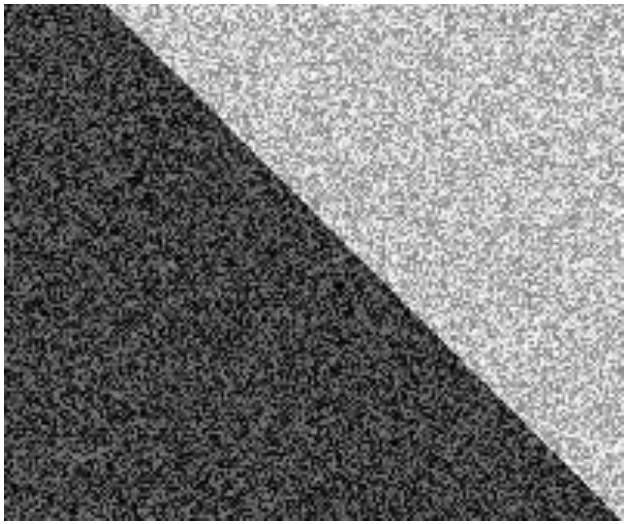
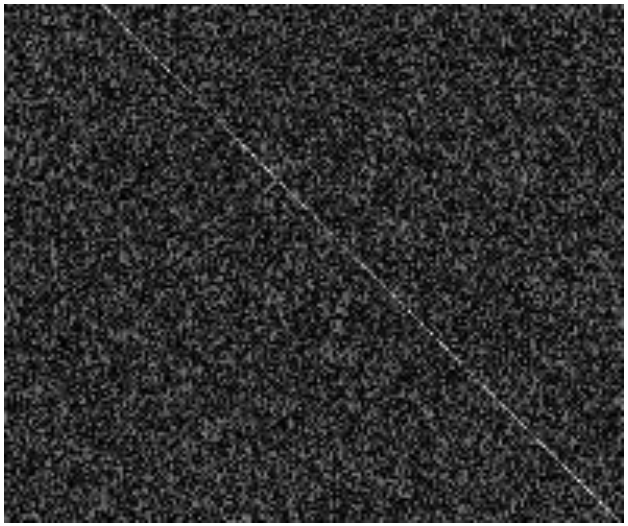
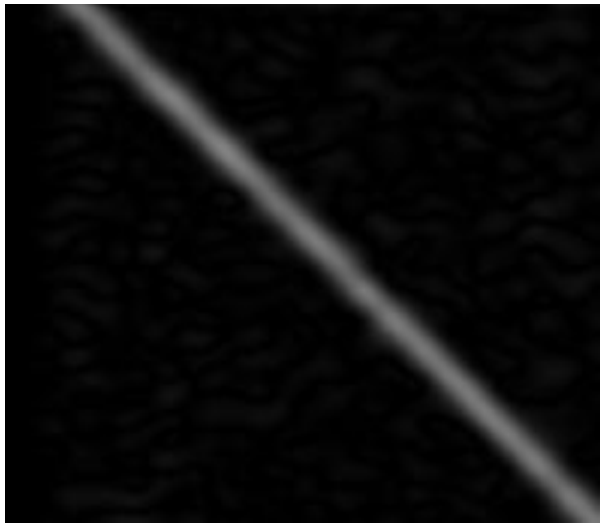


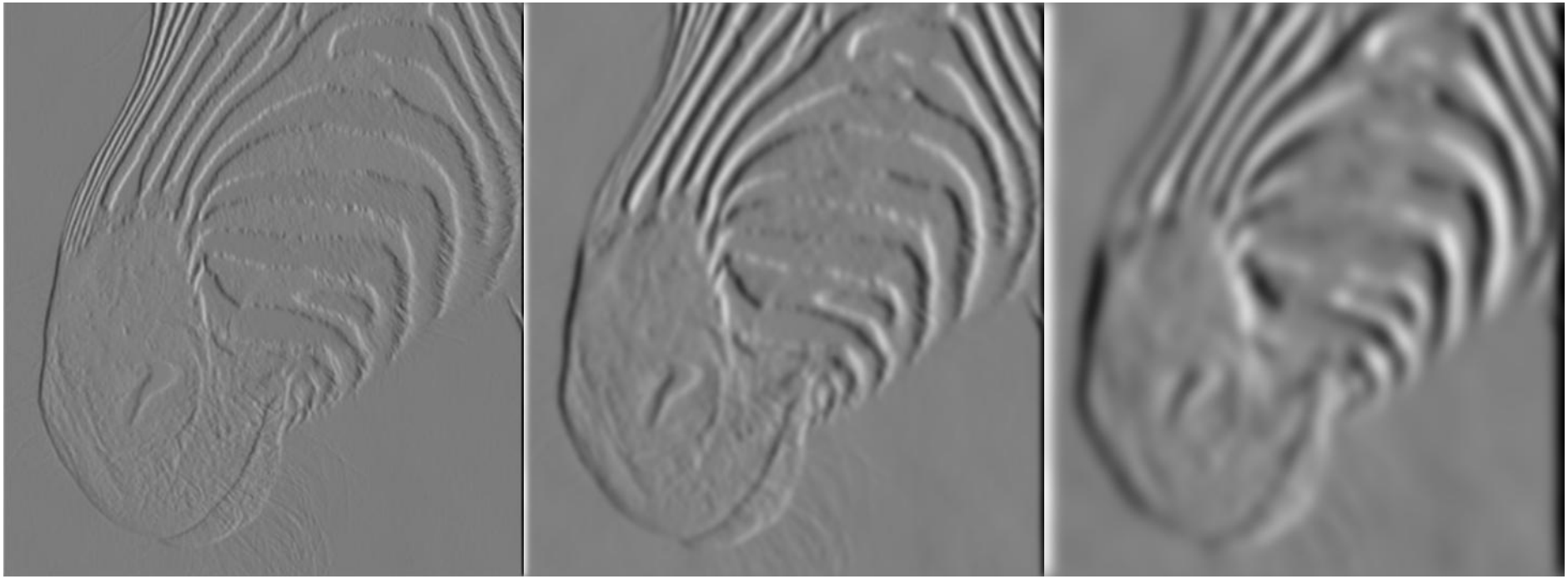
Image + Noise



Derivatives detect edge *and* noise



Smoothed derivative removes noise, but blurs edge



1 pixel

3 pixels

7 pixels

The scale of the smoothing filter affects derivative estimates

Corner Detection

Motivation: feature matching

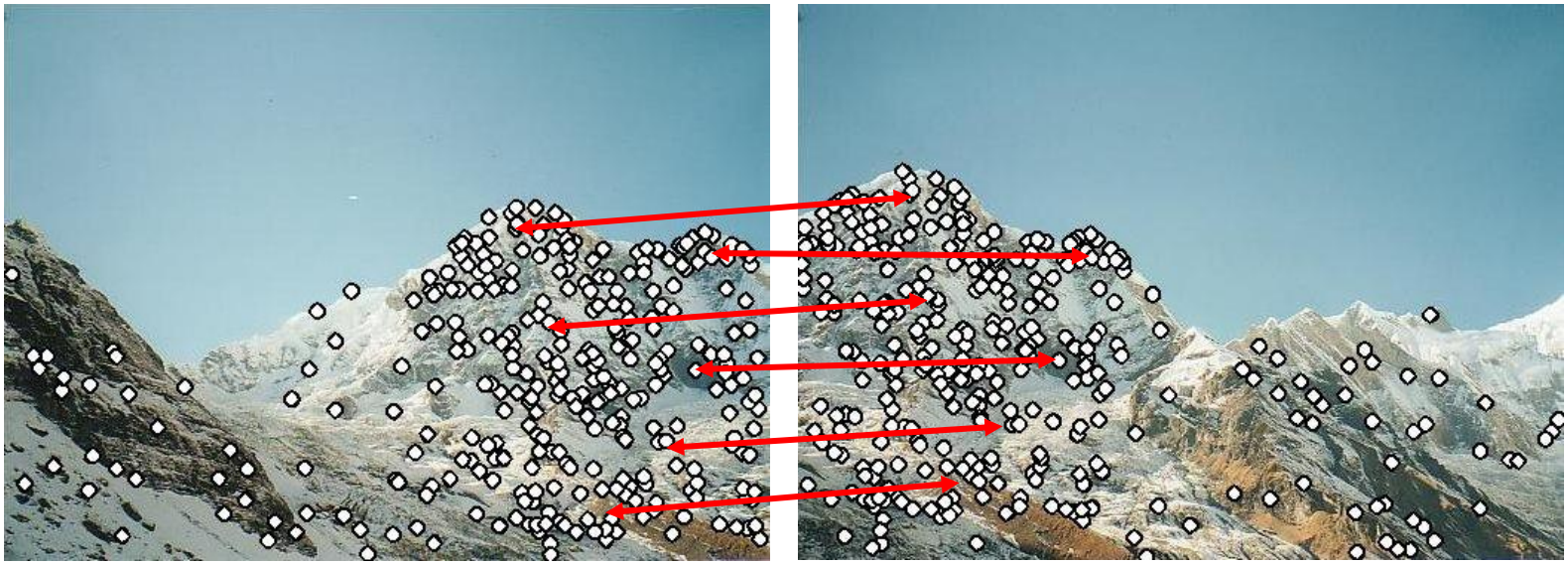
- Panorama stitching
  - We have two images – how do we combine them?





Motivation: feature matching

- Panorama stitching
  - We have two images – how do we combine them?



Step 1: extract features

Step 2: match features

Motivation: feature matching

- Panorama stitching
  - We have two images – how do we combine them?



Step 1: Extract features in each image

Step 2: Match features across images

Step 3: Align images and determine a transformation

Image matching



by Diva Sian



by swashford

Harder case

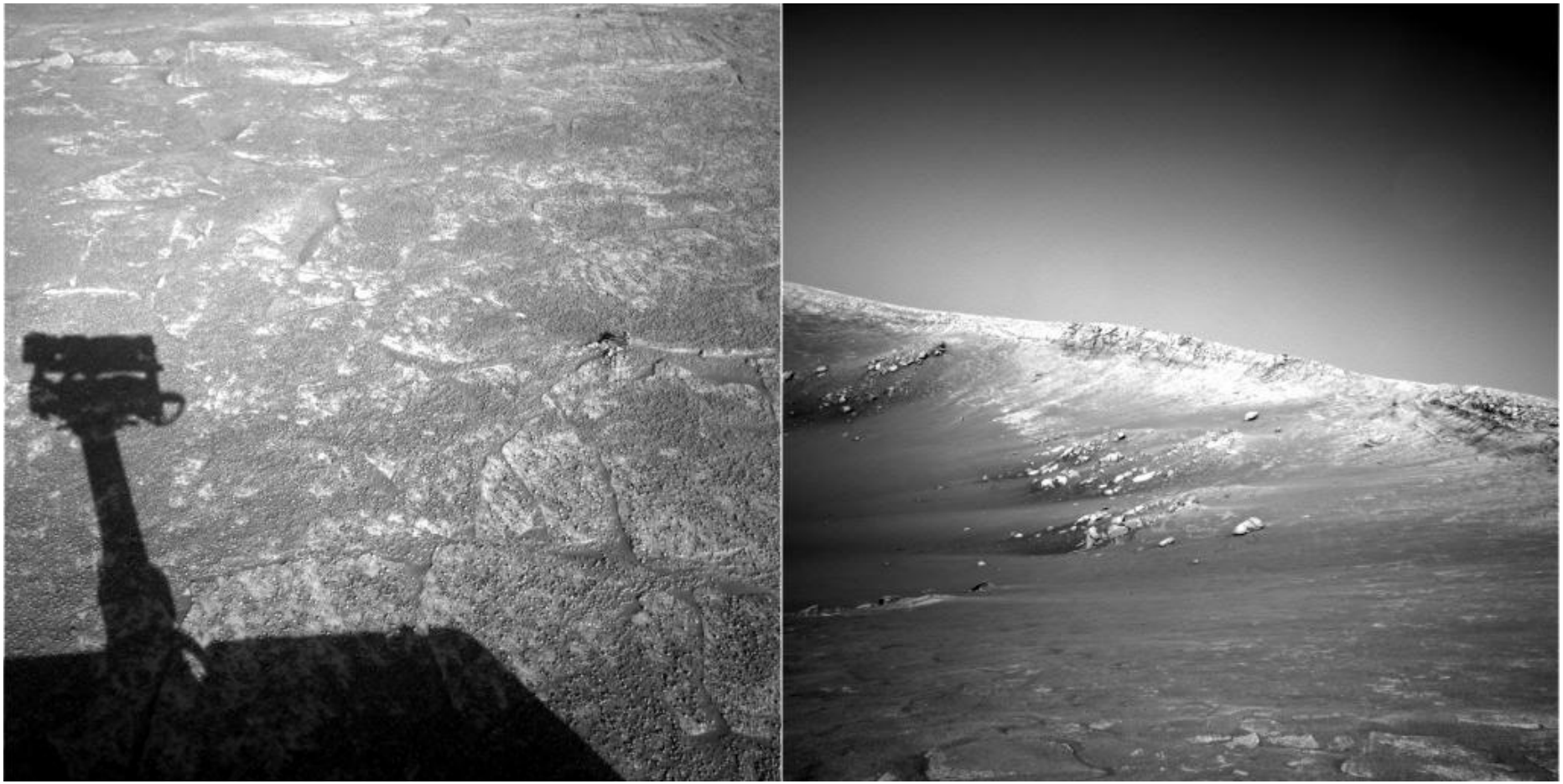


by Diva Sian



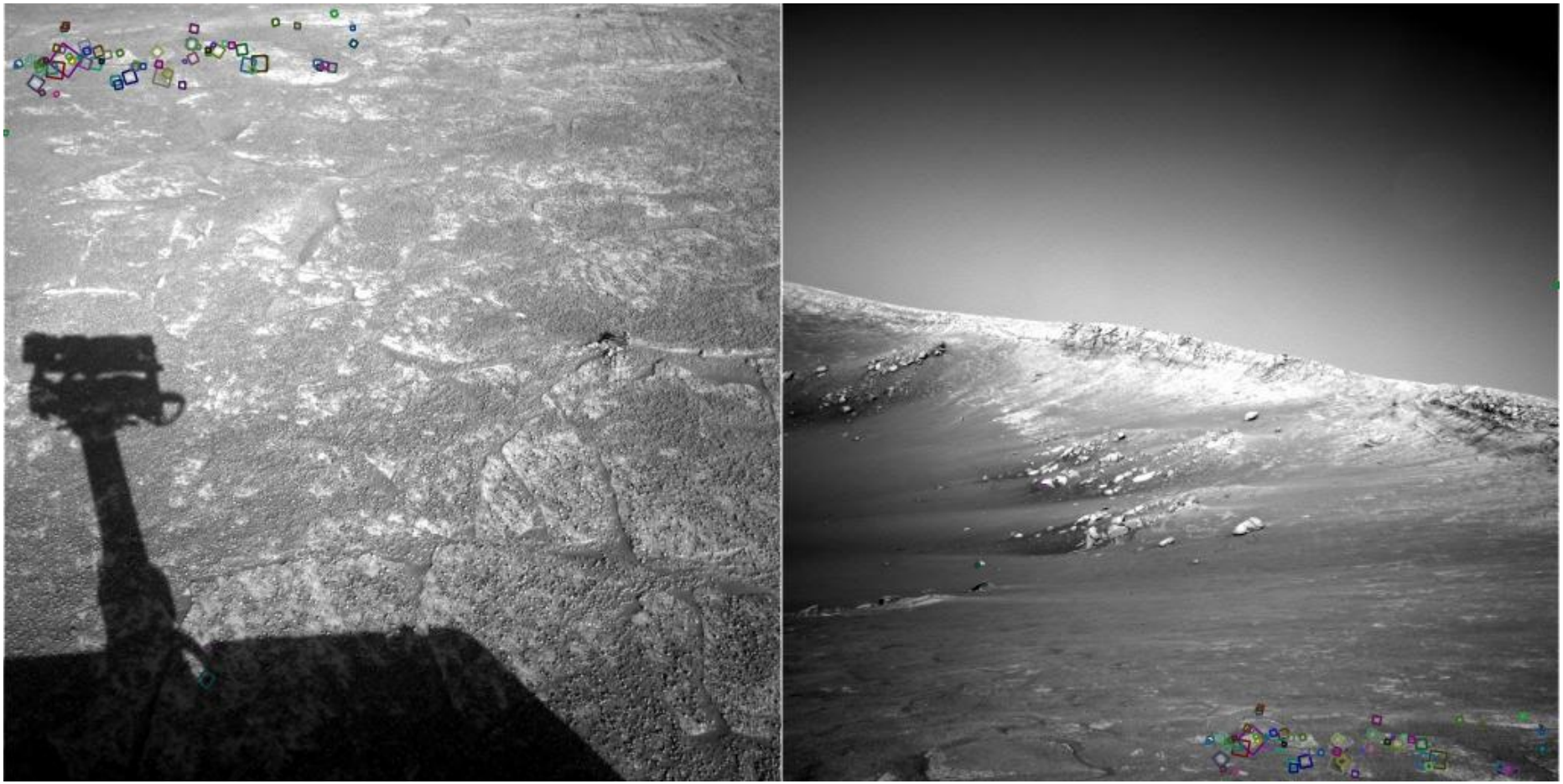
by scgbt

Harder still?



NASA Mars Rover images

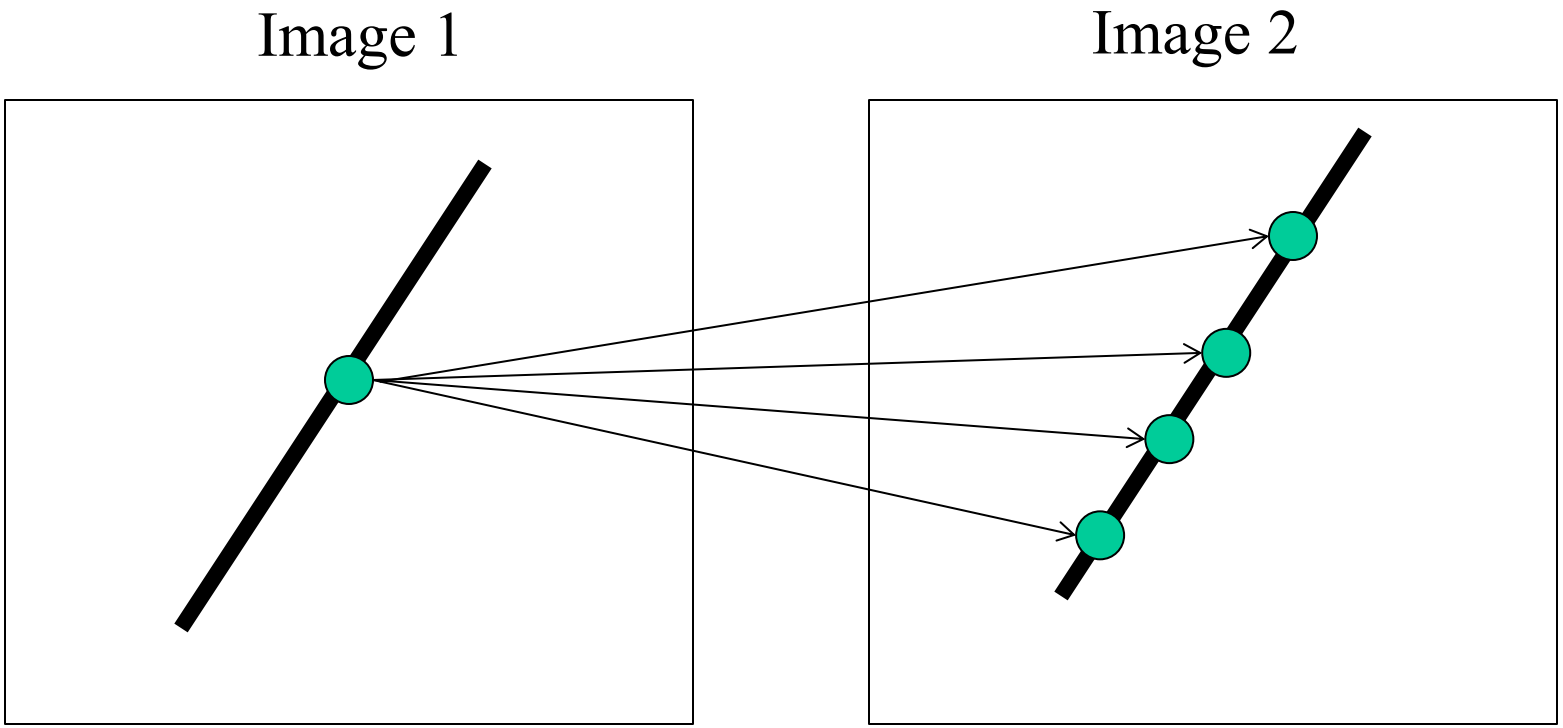
Answer below (look for tiny colored squares…)



NASA Mars Rover images
with SIFT feature matches
Figure by Noah Snavely

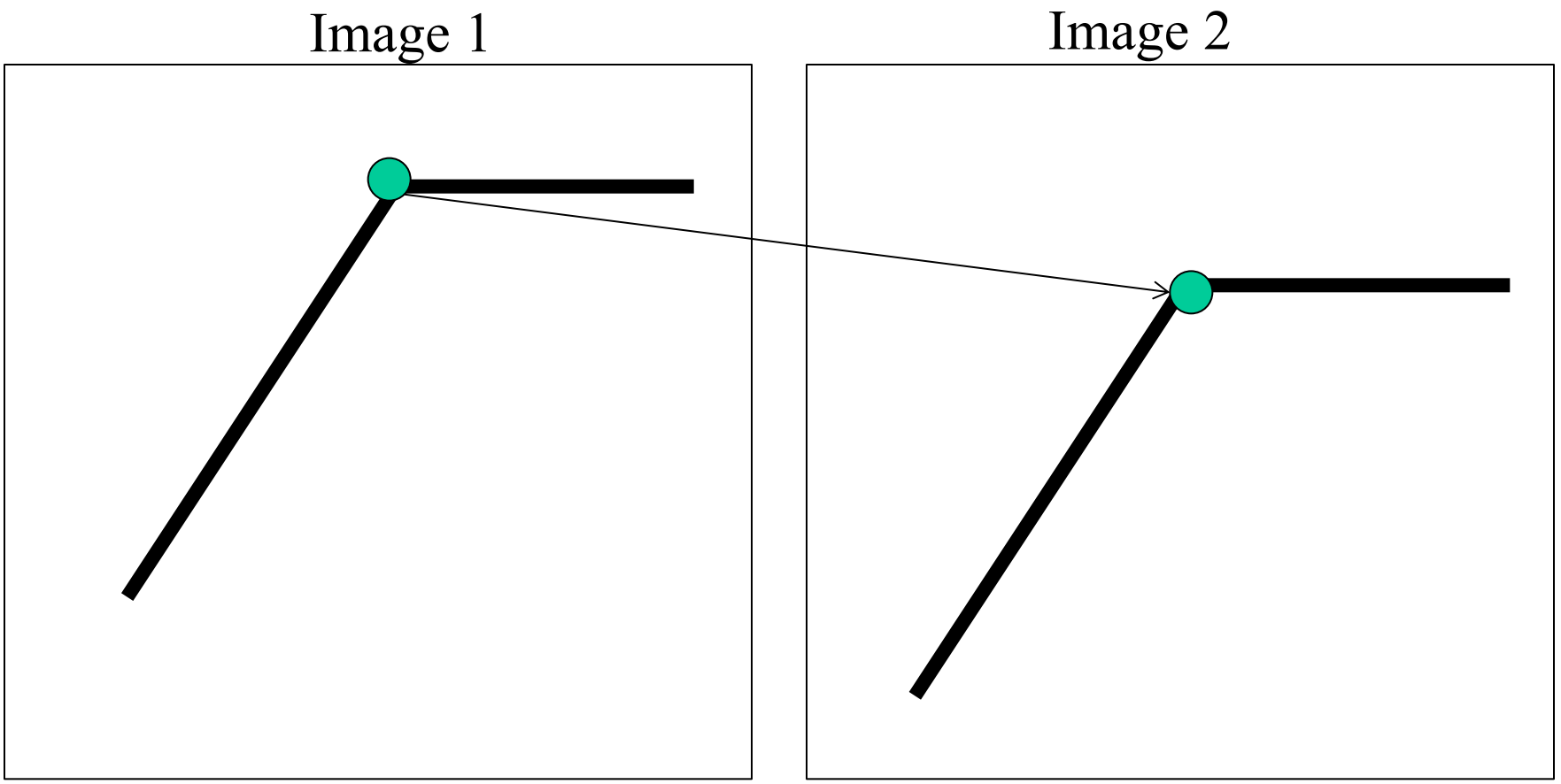
Corners contain more info than lines

- A point on a line is hard to match



Corners contain more info than lines

- A corner is easier to match

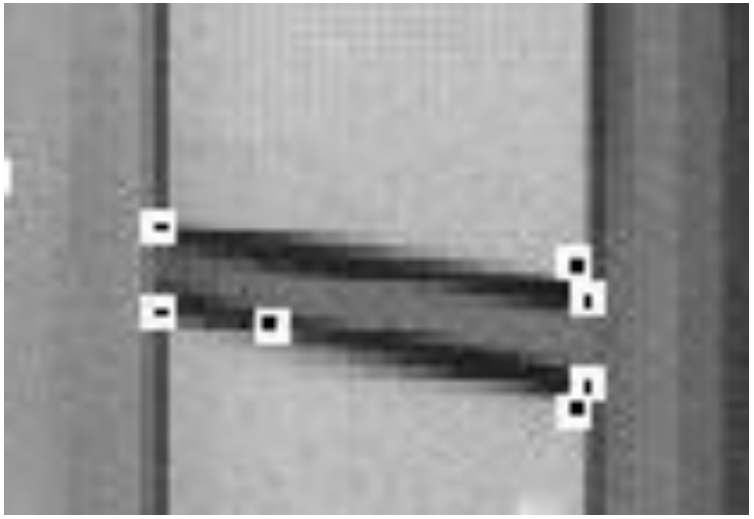


Detection of corner-like features

- Corner
  - A rapid change of direction in a curve
  - A highly effective feature
    - Distinctive
    - Reasonably invariant to viewpoint

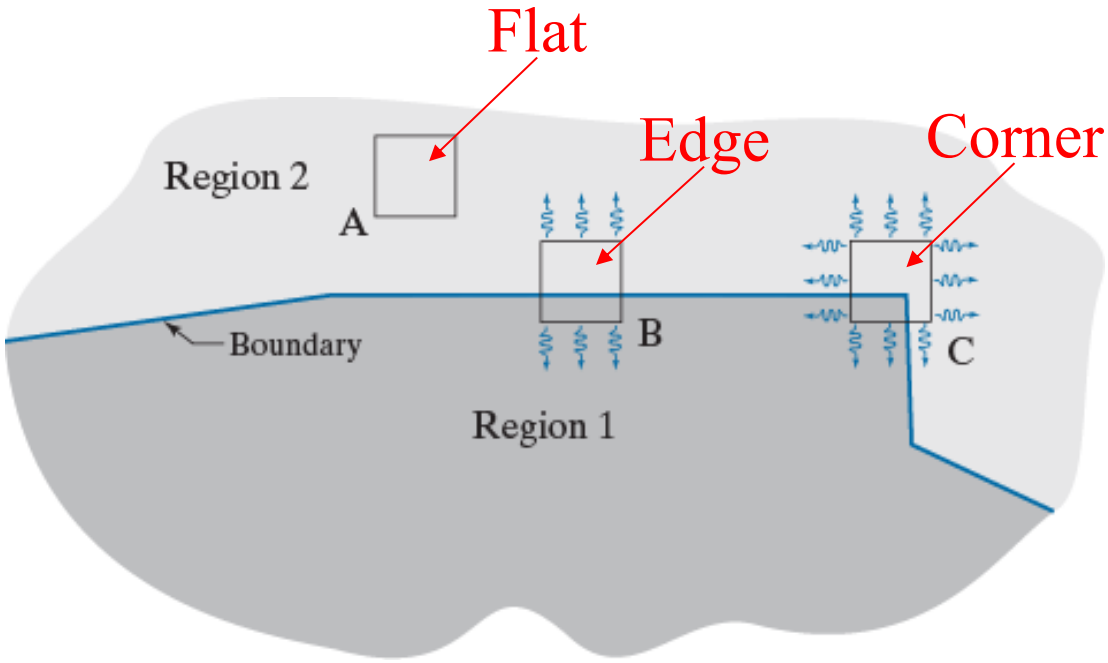
Corners





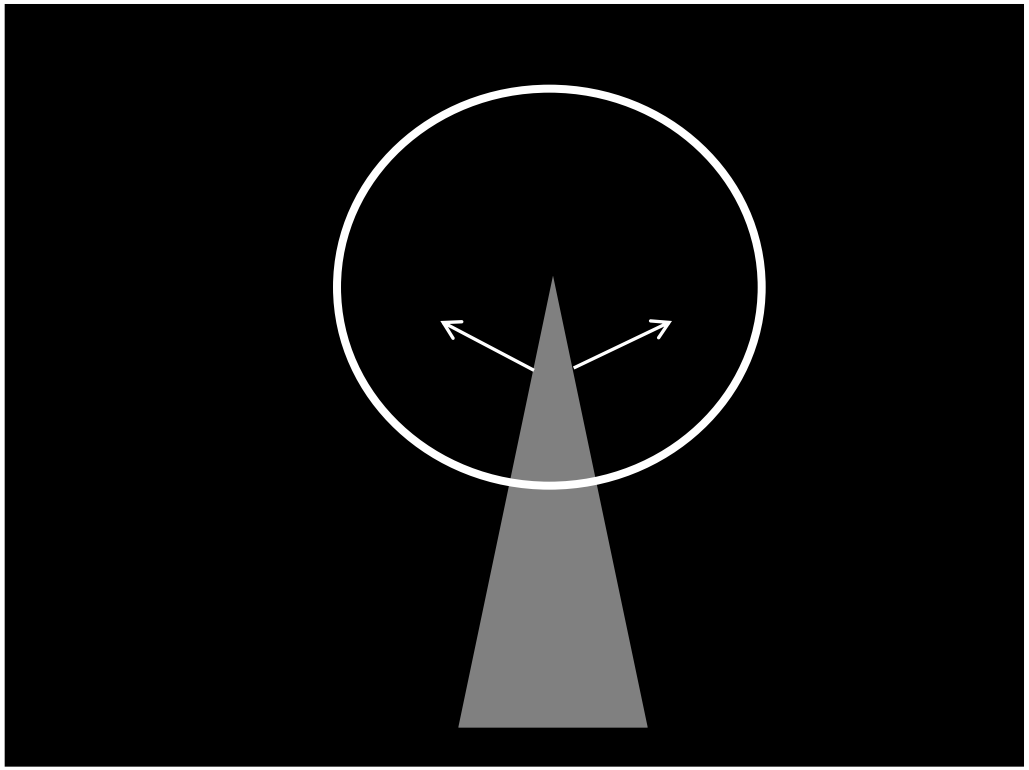
Detection of corner-like features

- Examine a small window over an image



The wiggly arrows indicate graphically a directional response in the detector as it moves in the three areas shown

Detection of corner-like features



Intuition:

- Right at corner, gradient is ill-defined.
- Near corner, gradient has two different values.

Detection of corner-like features

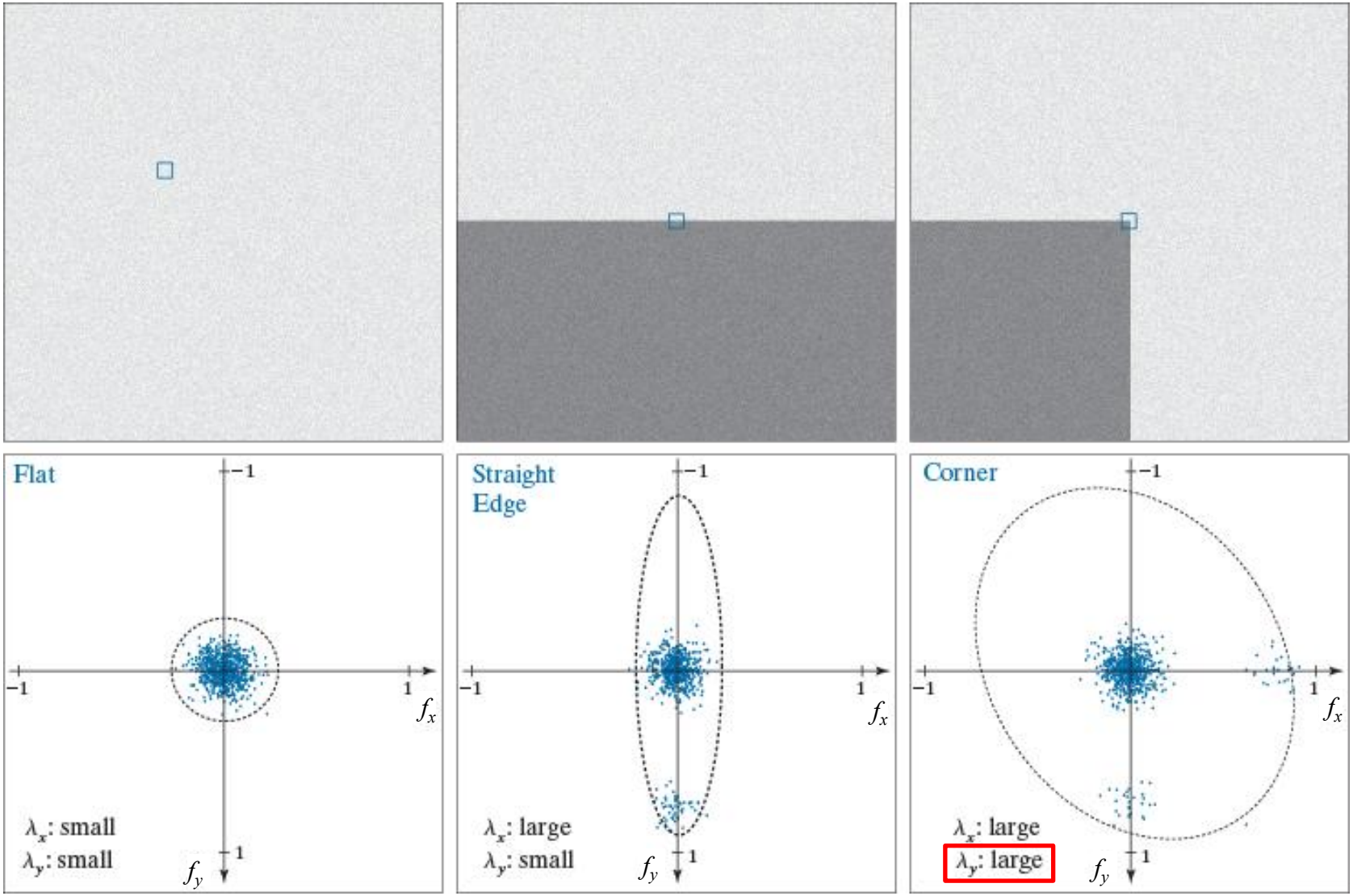
- For each window location, compute the spatial gradient matrix

$$\mathtt{M} = \begin{bmatrix} \sum_s \sum_t f_x^2(s,t) & \sum_s \sum_t f_x(s,t) f_y(s,t) \\ \sum_s \sum_t f_x(s,t) f_y(s,t) & \sum_s \sum_t f_y^2(s,t) \end{bmatrix}$$

  where f_x is the gradient in the x-direction and f_y is the gradient in the y-direction

- Then, compute eigenvalues of spatial gradient matrix

Eigenvalues of spatial gradient matrix



Detection of corner-like features

When computing gradients, use shape valid filtering followed by zero padding such that output image is same size as input image

- Shi-Tomasi corner detector
  - Run a small window over an image and compute spatial gradient matrix **M**
  - Compute the minor eigenvalue of **M** to compute measurement image R

$$\lambda_{\min} = \frac{\mathrm{Tr}(\mathtt{M}) - \sqrt{\mathrm{Tr}(\mathtt{M})^2 - 4\det(\mathtt{M})}}{2}$$

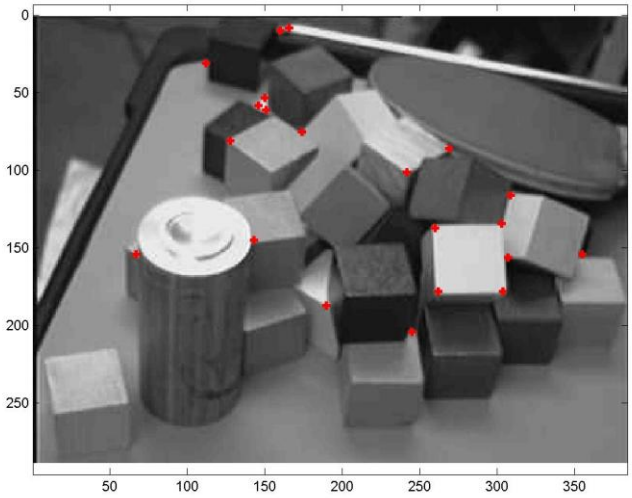
To mitigate floating-point numerical precision and accuracy issues, replace with

$$\sqrt{\max(0, \mathrm{Tr}(\mathtt{M})^2 - 4\det(\mathtt{M}))}$$

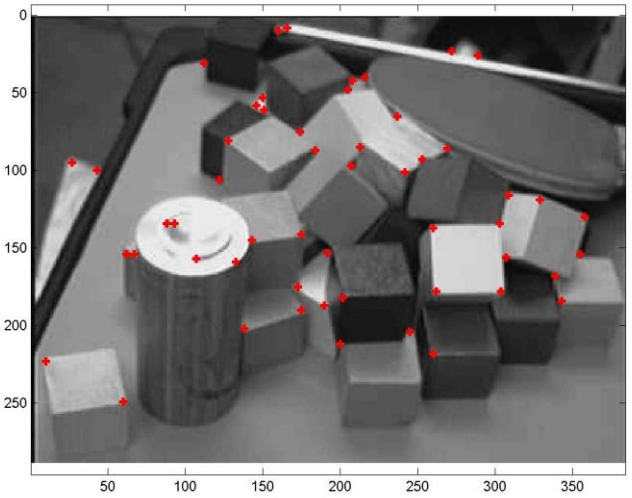
  - Apply nonmaximal suppression to the measurement image R
    - Prevents corners from being too close to each other
  - Threshold resulting image R using global threshold T
    - Corner at coordinates corresponding to $R > T$

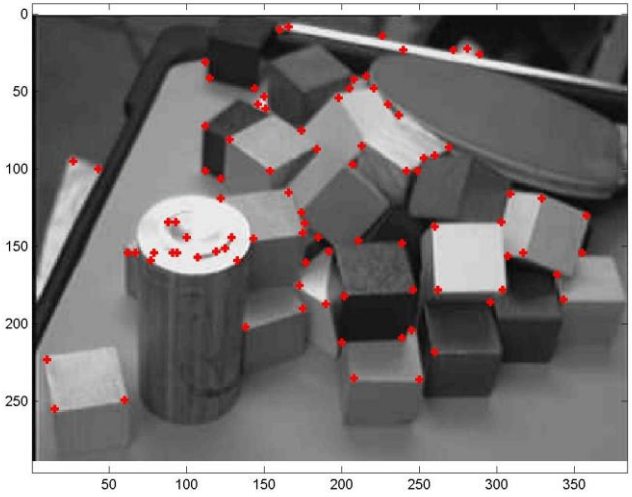
Corner Detection Sample Results

Threshold=25,000



Threshold=10,000





Threshold=5,000

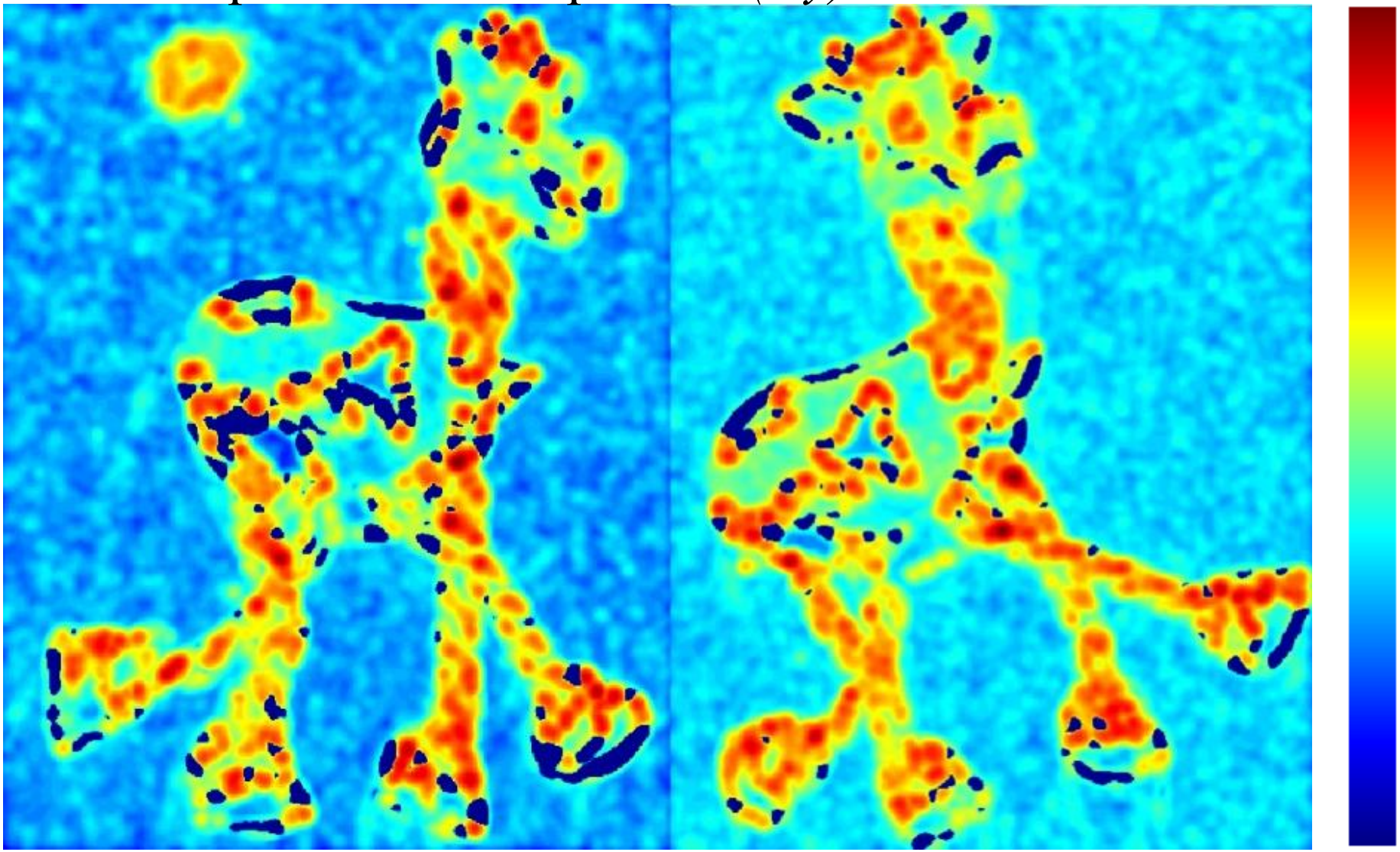
Corner Detector: Workflow



Slide credit: http://vims.cis.udel.edu/~chandra/

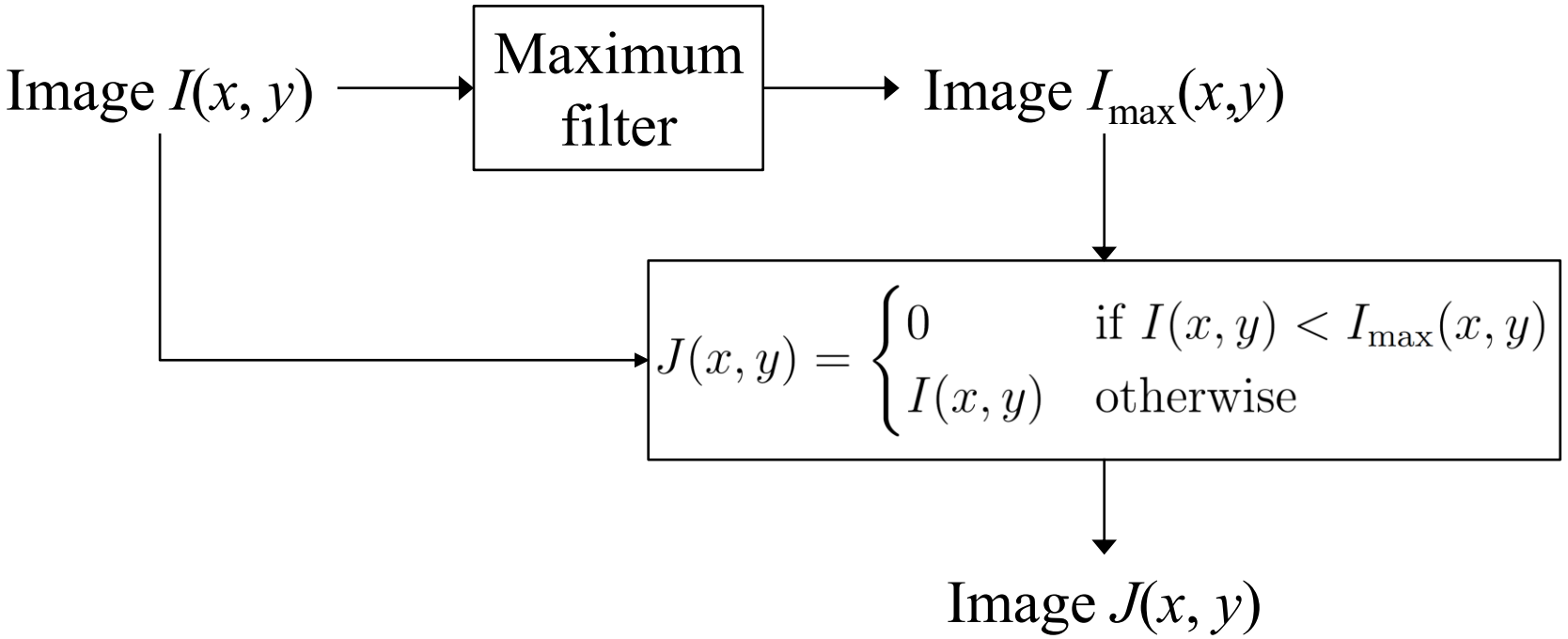
Corner Detector: Workflow

Compute corner response $R(x,y)$



Slide credit: http://vims.cis.udel.edu/~chandra/

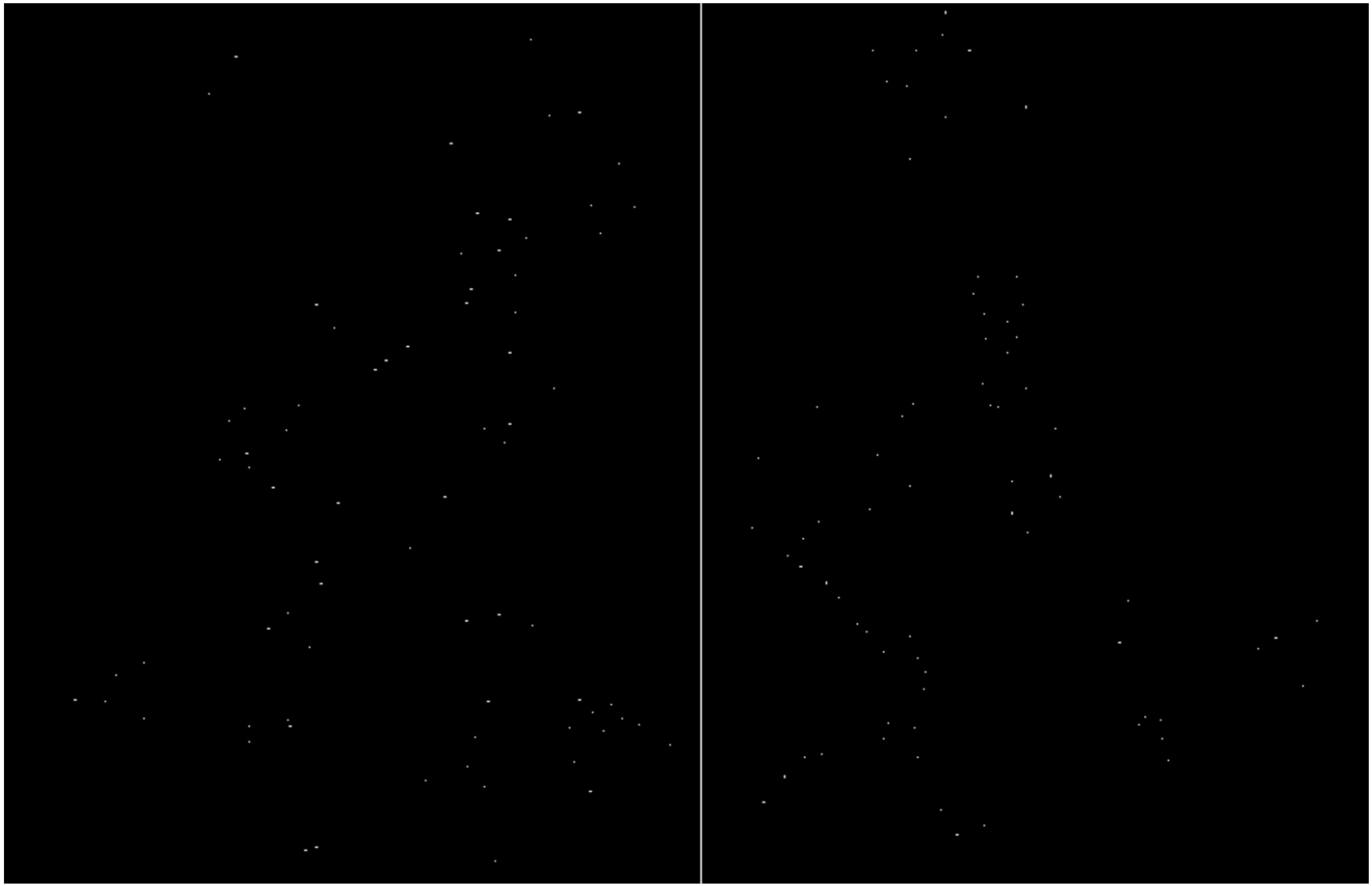
Nonmaximum suppression



- Then, find coordinates of pixels in image $J(x,y)$ with nonzero values

Corner Detector: Workflow

Take only the points of local maxima of $R(x,y)$ and threshold



Slide credit: http://vims.cis.udel.edu/~chandra/

Corner Detector: Workflow



Slide credit: http://vims.cis.udel.edu/~chandra/

Next Lecture

- Calibrated stereo