Edge and Corner Detection
Reading: Chapter 8 (skip 8.1)

- **Goal:** Identify sudden changes (discontinuities) in an image
- This is where most shape information is encoded
- **Example:** artist's line drawing (but artist is also using object-level knowledge)

What causes an edge?

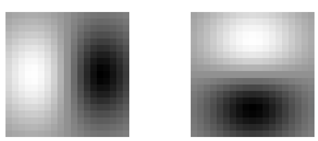
- Depth discontinuity
- Surface orientation discontinuity
- Reflectance discontinuity (i.e., change in surface material properties)
- Illumination discontinuity (e.g., shadow)

Slide credit: Christopher Rasmussen

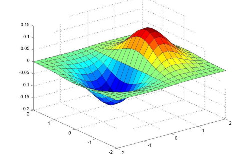
Smoothing and Differentiation

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

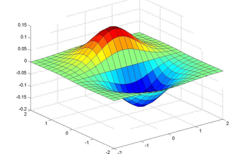
    D * (G * I) = (D * G) * I



Derivative of Gaussian



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



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

Gradient magnitude is computed from these.

Slide credit: Christopher Rasmussen

Gradient magnitude

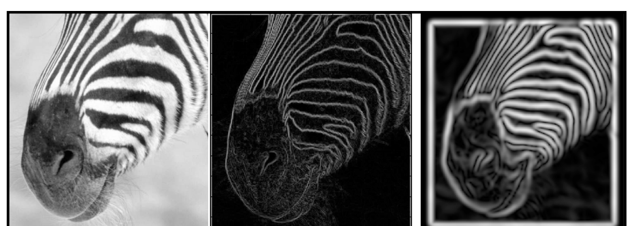
Let $I(X, Y)$ be a (digital) image

Let $I_x(X, Y)$ and $I_y(X, Y)$ be estimates of the partial derivatives in the x and y directions, respectively.

Call these estimates I_x and I_y (for short)

The vector $[I_x, I_y]$ is the gradient

The scalar $\sqrt{I_x^2 + I_y^2}$ is the gradient magnitude



Scale

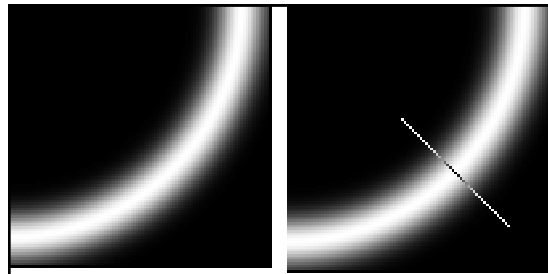
Increased smoothing:
- Eliminates noise edges.
- Makes edges smoother and thicker.
- Removes fine detail.

Canny Edge Detection

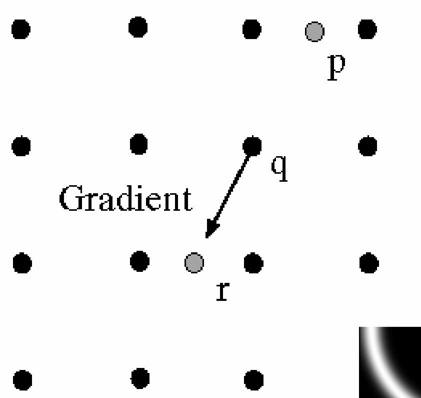
Steps:

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

- Matlab: `edge(I, ‘canny’)`

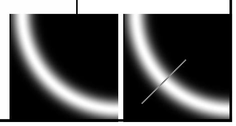


Non-maximum suppression:
Select the single maximum point across the width of an edge.



Non-maximum suppression

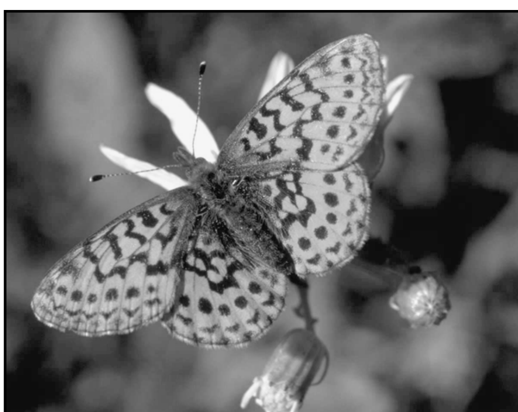
At q, the value must be larger than values interpolated at p or r.

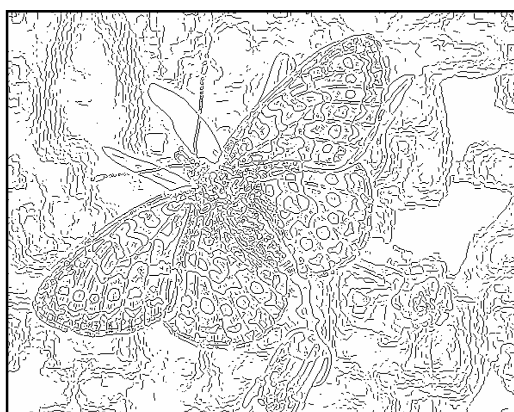


Examples:
Non-Maximum Suppression

Original image | Gradient magnitude | Non-maxima suppressed

Slide credit: Christopher Rasmussen

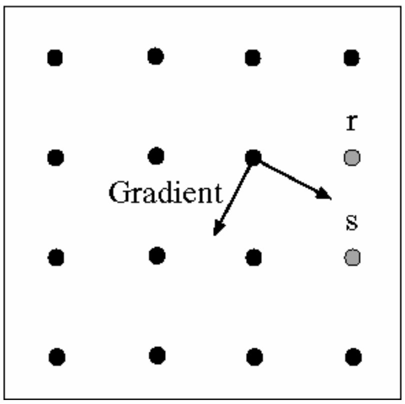




fine scale ($\sigma = 1$) high threshold

coarse scale, (σ = 4) 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).

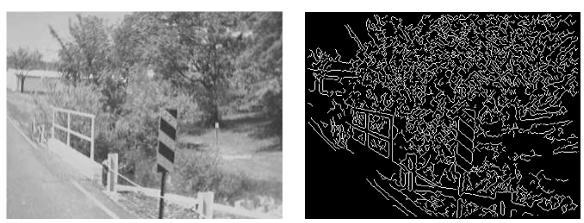
Edge Hysteresis

- **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 k_{low} to find weak edges which continue edge chain
- Typical ratio of thresholds is roughly

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

Example: Canny Edge Detection

Example: Canny Edge Detection



Using Matlab with default thresholds

Slide credit: Christopher Rasmussen

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.

The Harris corner detector

Form the second-moment matrix:

Sum over a small region around the hypothetical corner

Gradient with respect to x, times gradient with respect to y

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

Matrix is symmetric

Slide credit: David Jacobs

Simple Case

First, consider case where:

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

This means dominant gradient directions align with x or y axis

If either λ is close to 0, then this is **not** a corner, so look for locations where both are large.

Slide credit: David Jacobs

General Case

It can be shown that since C is symmetric:

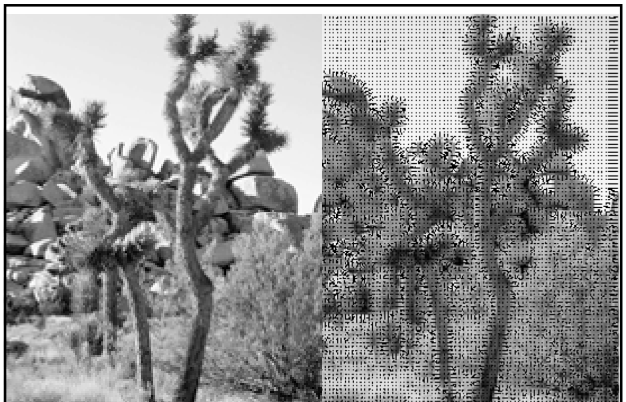
$$C = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

So every case is like a rotated version of the one on last slide.

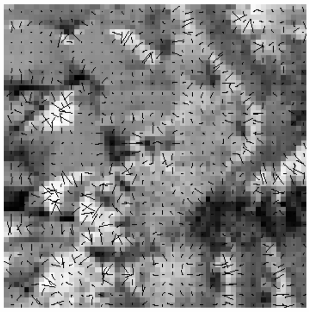
Slide credit: David Jacobs

So, to detect corners

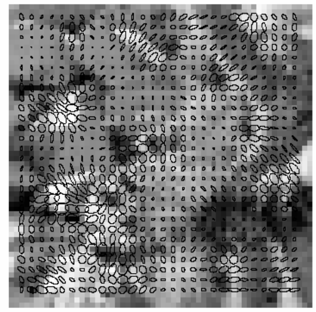
- Filter image with Gaussian to reduce noise
- Compute magnitude of the x and y gradients at each pixel
- Construct C in a window around each pixel (Harris uses a Gaussian window – just blur)
- Solve for product of λs (determinant of C)
- If λs are both big (product reaches local maximum and is above threshold), we have a corner (Harris also checks that ratio of λs is not too high)



Gradient orientations

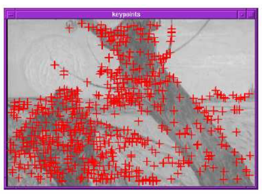
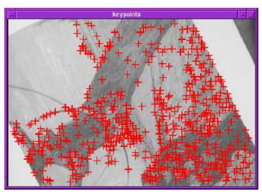


Closeup of gradient orientation at each pixel



Corners are detected where the product of the ellipse axis lengths reaches a local maximum.

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)