Edges

What is an edge?
A discontinuity in image intensity.

Physical causes of edges
1. Object boundaries
2. Surface normal discontinuities
3. Reflectance (albedo) discontinuities
4. Lighting discontinuities (shadow boundaries)

Corners

Object Boundaries
Surface normal discontinuities

Boundaries of materials properties

Boundaries of lighting

Profiles of image intensity edges

Noisy 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, derivative has maximum magnitude
• Or 2nd derivative is zero.
Numerical Derivatives

Take Taylor series expansion of \( f(x) \) about \( x_0 \):
\[
\begin{align*}
\quad\quad f(x) &= f(x_0) + f'(x_0)(x-x_0) + \frac{1}{2} f''(x_0)(x-x_0)^2 + \cdots \\
\text{Consider samples taken at increments of } h \text{ and first two terms of the expansion, we have:} \\
\quad\quad f(x_0+h) &= f(x_0) + f'(x_0)h + \frac{1}{2} f''(x_0)h^2 \\
\quad\quad f(x_0-h) &= f(x_0) - f'(x_0)h + \frac{1}{2} f''(x_0)h^2
\end{align*}
\]
Subtracting and adding \( f(x_0+h) \) and \( f(x_0-h) \) respectively yields:
\[
\begin{align*}
\frac{f'(x_0)}{2h} &= \frac{f(x_0+h) - f(x_0-h)}{2h} \\
\frac{f''(x_0)}{h^2} &= \frac{f(x_0+h) - 2f(x_0) + f(x_0-h)}{h^2}
\end{align*}
\]

Implementing 1-D Edge Detection

1. Filter out noise: convolve with Gaussian
2. Take a derivative: convolve with \([-1/2 0 1/2]\)
   - We can combine 1 and 2.
3. Find the peak: Two issues:
   - Should be a local maximum.
   - Should be sufficiently high.

Canny Edge Detector

1. Smooth image by filtering with a Gaussian
2. Compute gradient at each point in the image.
3. At each point in the image, compute the direction of the gradient and the magnitude of the gradient.
4. Perform non-maximal suppression to identify candidate edgels.
5. Trace edge chains using hysteresis thresholding.

Gradient

- Given a function \( f(x,y) \) -- e.g., intensity is \( f \)
- Gradient equation:
\[
\nabla f = \left[ \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]
\]
- Represents direction of most rapid change in intensity
- Gradient direction:
\[
\theta = \tan^{-1} \left( \frac{\partial f}{\partial y} / \frac{\partial f}{\partial x} \right)
\]
- The edge strength is given by the gradient magnitude
\[
||\nabla f|| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}
\]
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!

The scale of the smoothing filter affects derivative estimates

Non-maximum suppression

Loop over every point \( q \) in the image, decide whether \( q \) is a candidate edge point
Using gradient direction at \( q \), find two points \( p \) and \( r \) on adjacent rows (or columns).
If \( |\nabla f(q)| > |\nabla f(p)| \) and \( |\nabla f(q)| > |\nabla f(r)| \)
then \( q \) is a candidate edge point

Non-maximum suppression

Loop over every point \( q \) in the image, decide whether \( q \) is a candidate edge point
Using gradient direction at \( q \), find two points \( p \) and \( r \) on adjacent rows (or columns).
If \( |\nabla f(q)| > |\nabla f(p)| \) and \( |\nabla f(q)| > |\nabla f(r)| \)
then \( q \) is a candidate edge point
An Idea: Single Threshold

1. Smooth Image
2. Compute gradients & Magnitude
3. Non-maximal suppression
4. Compare to a threshold: T

A Better Idea: Linking + Two Thresholds

Linking: Assume the marked point q is an edge point. Then we construct the tangent to the edge curve (which is normal to the gradient at that point) and use this to predict the next points (either r or s).
A Better Idea: Hysteresis Thresholding

- Define two thresholds $\tau_{\text{low}}$ and $\tau_{\text{high}}$.
- Starting with output of nonmaximal suppression, find a point $q_0$ where $\nabla^2 I(q_0) > \tau_{\text{sup}}$ and $\nabla I(q_0)$ is a local maximum.
- Start tracking an edge chain at pixel location $q_0$ in one of the two directions.
- Stop when gradient magnitude $< \tau_{\text{low}}$.
  - i.e., use a high threshold to start edge curves and a low threshold to continue them.

![Graph showing gradient magnitude and thresholds](image)

Single Threshold

- $T = 15$
- $T = 5$

Hysteresis

- $T_h = 15$
- $T_l = 5$

Hysteresis thresholding

![Butterfly image](image)

- Fine scale, high threshold
- Coarse scale, high high threshold

![Coarse scale, low high threshold image](image)
Why is Canny so Dominant

- Still widely used after 29 years.
  1. Theory is nice
  2. Details good (magnitude of gradient, non-max suppression).
  3. Hysteresis an important heuristic.
  4. Code was distributed.

Corner Detection

Feature extraction: Corners

Why extract features?

- Motivation: panorama stitching
  - We have two images – how do we combine them?

Step 1: extract features
Step 2: match features

Why extract features?

- Motivation: panorama stitching
  - We have two images – how do we combine them?

Step 1: extract features
Step 2: match features
Step 3: align images
Corners contain more info than lines.

- A point on a line is hard to match.

The Basic Idea

- We should easily recognize the point by looking through a small window
- Shifting a window in any direction should give a large change in intensity

Edge Detectors Tend to Fail at Corners

Finding Corners

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

Distribution of gradients for different image patches
Finding Corners

For each image location \((x,y)\), we create a matrix \(C(x,y)\):

\[
C(x,y) = \begin{bmatrix}
\sum I_x^2 & \sum I_x I_y \\
\sum I_x I_y & \sum I_y^2
\end{bmatrix}
\]

Gradient with respect to \(x\), times gradient with respect to \(y\)
Sum over a small region

Matrix is symmetric

WHY THIS?

Because \(C\) is a symmetric positive definite matrix, it can be factored as:

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

where \(R\) is a 2x2 rotation matrix and \(\lambda_1\) and \(\lambda_2\) are non-negative.

1. \(\lambda_1\) and \(\lambda_2\) are the Eigenvalues of \(C\).
2. The columns of \(R\) are the Eigenvectors of \(C\).
3. Eigenvalues can be found by solving the characteristic equation \(\det(C - \lambda I) = 0\) for \(\lambda\).

Example: Assume \(R=\text{Identity (axis aligned)}\)

What is region like if:
1. \(\lambda_1 = 0\), \(\lambda_2 > 0\)?
2. \(\lambda_2 = 0\), \(\lambda_1 > 0\)?
3. \(\lambda_1 = 0\) and \(\lambda_2 = 0\)?
4. \(\lambda_1 >> 0\) and \(\lambda_2 >> 0\)?

Defining the "Corner Response"

\[
\text{Corner} \Leftrightarrow \max \{\lambda_1, \lambda_2\} \neq 0
\]

Corner Detection Sample Results

Threshold=25,000
Threshold=10,000
Threshold=5,000

So, to detect corners

- Filter image with a Gaussian.
- Compute the gradient everywhere.
- Move window over image, and for each window location:
  1. Construct the matrix \(C\) over the window.
  2. Use linear algebra to find \(\lambda_1\) and \(\lambda_2\).
  3. If they are both big, we have a corner.

1. Let \(e(x,y) = \min(\lambda_1(x,y), \lambda_2(x,y))\)
2. \((x,y)\) is a corner if it’s local maximum of \(e(x,y)\) and \(e(x,y) > \tau\)

Parameters: Gaussian std. dev., window size, threshold