Edge detection

- **Goal**: Identify sudden changes (discontinuities) in an image.
  - Intuitively, most semantic and shape information from the image can be encoded in the edges.
  - More compact than pixels.
- **Ideal**: artist’s line drawing (artist also uses object-level knowledge)
• Edges are caused by a variety of factors

Source: Steve Seitz

C. Nikou – Image Analysis (T-14)
Edge model (cont.)

Edge point detection
• Magnitude of the first derivative.
• Sign change of the second derivative.

Observations:
• Second derivative produces two values for an edge (undesirable).
• Its zero crossings may be used to locate the centres of thick edges.

Image gradient

• The gradient of an image: \( \nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \)

\[ \nabla f = [\frac{\partial f}{\partial x}, 0] \quad \nabla f = [0, \frac{\partial f}{\partial y}] \]

The gradient points in the direction of most rapid increase in intensity

The gradient direction is given by \( \theta = \tan^{-1} \left( \frac{\frac{\partial f}{\partial y}}{\frac{\partial f}{\partial x}} \right) \)

• how does this relate to the direction of the edge?

The edge strength is given by the gradient magnitude

\[ \| \nabla f \| = \sqrt{\left( \frac{\partial f}{\partial x} \right)^2 + \left( \frac{\partial f}{\partial y} \right)^2} \]
Differentiation and convolution

• Recall, for 2D function, \( f(x,y) \):

\[
\frac{\partial f}{\partial x} = \lim_{\varepsilon \to 0} \left( \frac{f(x + \varepsilon, y) - f(x, y)}{\varepsilon} \right)
\]

This is linear and shift invariant. It is therefore the result of a convolution.

• Approximation:

\[
\frac{\partial f}{\partial x} \approx \frac{f(x_{n+1}, y) - f(x_n, y)}{\Delta x}
\]

which is a convolution with the mask:

\[-1 \ 1\]

Finite difference filters

• Other approximations of derivative filters:

<table>
<thead>
<tr>
<th>-1</th>
<th>0</th>
<th>0</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Roberts

<table>
<thead>
<tr>
<th>-1</th>
<th>-1</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Prewitt

<table>
<thead>
<tr>
<th>-1</th>
<th>-2</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

Sobel

Source: D. Forsyth, D. Lowe

Source: K. Grauman
Effect of Finite Differences on White Noise

Let \( f \) be an observed instance of the image \( f_0 \) corrupted by noise \( w \):

\[
f = f_0 + w
\]

with noise samples having mean value \( E[w(n)] = 0 \) and being uncorrelated with respect to location:

\[
E[w(m)w(n)] = \begin{cases} 
\sigma^2, & m = n \\
0, & m \neq n 
\end{cases}
\]
Effect of Finite Differences on White Noise (cont.)

Applying a filter $h$ to the degraded image:

$$g = h \ast f = h \ast (f_0 + w) = h \ast f_0 + h \ast w$$

The expected value of the output is:

$$E[g] = E[h \ast f_0] + E[h \ast w] = h \ast f_0 + h \ast E[w]$$

$$= h \ast f_0 + h \ast 0 = h \ast f_0$$

The noise is removed in average.

Effect of Finite Differences on White Noise (cont.)

What happens to the standard deviation of $g$?

$$\sigma^2_g = \sigma^2 \sum_{i=1}^{N} h^2(k)$$

Assume that a finite difference filter is used, which approximates the first derivative:

$$\begin{bmatrix}
-1 \\
1
\end{bmatrix}$$

$$\sigma^2_g = 2\sigma^2$$
For a second derivative approximation:

\[
\begin{bmatrix}
0 & 0 & 0 \\
1 & -2 & 1 \\
0 & 0 & 0 \\
\end{bmatrix}
\]

\[\sigma_g^2 = 6\sigma^2\]

The standard deviation increases sharply! Higher order coefficients form a Pascal triangle.

Another perspective: differentiation in the Fourier domain:

\[
\frac{d}{dx} f(x) \leftrightarrow j\Omega F(\Omega)
\]

High frequencies are heavily emphasized with respect to low spatial frequencies.
Edge model and noise

Intensity profile  First derivative  Second derivative

Edge model and noise (cont.)

Intensity profile  First derivative  Second derivative
• Edge detectors respond strongly to sharp changes and noise is a sharp change.
• As we have seen, simple finite differences are unusable.
• Noise is sometimes useful!
  – Pictures in a dark room with the lens cap on helps to determine information about the camera temperature.

• Independent additive stationary Gaussian noise is the simplest model.
• Issues
  – this model allows noise values that could be greater than maximum camera output or less than zero.
  – for small standard deviations, this isn’t too much of a problem - it’s a fairly good model.
  – independence may not be justified (e.g. damage to lens).
  – may not be stationary (e.g. thermal gradients in the ccd).
Why smoothing helps

Smoothing (e.g. average) kernels have

\[ \sum_{i=1}^{N} \sum_{j=1}^{N} h(i, j) = 1 \iff \sum_{i=1}^{N} \sum_{j=1}^{N} h^2(i, j) \leq 1 \]

Therefore, the noise standard deviation is reduced at the output, as we saw before.

Also, after smoothing (e.g. average), noise samples are no longer independent. They look like their neighbors and the derivatives are smaller.

Application of noise smoothing

In computer graphics, noise smoothing is widely used as it is a source of representing natural textures (e.g. smoke).
Noisy Edge Smoothing

- To find edges, look for peaks in $\frac{d}{dx}(f \ast g)$.

Convolution property

- Convolve with the derivative of the filter.

Source: S. Seitz
Derivative of Gaussian

\[ [1 \ -1] \]

Derivative of Gaussian (cont.)

x-direction

y-direction
Tradeoff between smoothing and localization

Smoothed derivative removes noise but blurs edge. It also finds edges at different "scales" affecting the semantics of the edge.

Source: D. Forsyth

The LoG operator

A good place to look for edges is the maxima of the first derivative or the zeros of the second derivative. The 2D extension approximates the second derivative by the Laplacian operator (which is rotationally invariant):

\[ \nabla^2 f(x, y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \]

Marr and Hildreth [1980] argued that a satisfactory operator that could be tuned in scale to detect edges is the Laplacian of the Gaussian (LoG).

The LoG operator (cont.)

\[ \nabla^2 \left( G(x, y) * f(x, y) \right) = \nabla^2 G(x, y) * f(x, y) \]

- The LoG operator is given by:

\[
\nabla^2 G(x, y) = \frac{\partial^2 G}{\partial x^2} + \frac{\partial^2 G}{\partial y^2} = \left( \frac{x^2 + y^2 - 2\sigma^2}{\sigma^4} \right) e^{\frac{-x^2+y^2}{2\sigma^2}}
\]

- The zero crossings of the operator indicate edge pixels. These may be computed, for instance, by using a 3x3 window around a pixel and detect if two of its opposite neighbors have different signs (and their difference is significant compared to a threshold) in result of the LoG operator.

C. Nikou – Image Analysis (T-14)
The LoG operator (cont.)

Image LoG

Zero crossings with a threshold of 4% of the image max

Contrast threshold = 1

Contrast threshold = 4

σ = 2

σ = 4
• Filter the image at various scales and keep the zero crossings that are common to all responses.

• Marr and Hildreth [1980] showed that LoG may be approximated by a difference of Gaussians (DOG):

\[
\text{DoG}(x,y) = \frac{1}{2\pi\sigma_1^2} e^{-\frac{x^2+y^2}{2\sigma_1^2}} - \frac{1}{2\pi\sigma_2^2} e^{-\frac{x^2+y^2}{2\sigma_2^2}}, \quad \sigma_1 > \sigma_2
\]

• Certain channels of the human visual system are selective with respect to orientation and frequency and can be modeled by a DoG with a ratio of standard deviations of 1.75.

• Meaningful comparison between LoG and DoG may be obtained after selecting the value of \( \sigma \) for LoG so that LoG has the same zero crossings as DoG:

\[
\sigma^2 = \frac{\sigma_1^2 \sigma_2^2}{\sigma_1^2 - \sigma_2^2} \ln \left( \frac{\sigma_1^2}{\sigma_2^2} \right)
\]

• The two functions should also be scaled to have the same value at the origin.
The LoG operator (cont.)

- LoG has fallen to some disfavour.
- It is not oriented and its response averages the responses across the two directions.
  - Poor response at corners.
  - Difficulty in recording the topology of T-junctions (trihedral vertices).
- Several studies showed that image components along an edge contribute to the response of LoG to noise but not to a true edge. Thus, zero crossings may not lie exactly on an edge.

Poor corner and trihedral vertices detection
Problems with gradient-based edge detectors

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 thick trail; how do we identify only the significant points?
3. How do we link the relevant points up into curves?

Designing an optimal edge detector

- Criteria for an "optimal" edge detector [Canny 1986]:
  - **Good detection**: the optimal detector must minimize the probability of false positives (detecting spurious edges caused by noise), as well as that of false negatives (missing real edges).
  - **Good localization**: the edges detected must be as close as possible to the true edges
  - **Single response**: the detector must return one point only for each true edge point; that is, minimize the number of local maxima around the true edge.
Canny edge detector

- Probably the most widely used edge detector in computer vision.
- Theoretical model: step-edges corrupted by additive Gaussian noise.
- Canny has shown that the first derivative of the Gaussian closely approximates the operator that optimizes the product of signal-to-noise ratio and localization.


Canny edge detector (cont.)

1. Filter image with derivative of Gaussian.
2. Find magnitude and orientation of gradient.
3. Non-maximum suppression:
   - Thin multi-pixel wide “ridges” down to single pixel width.
4. Linking and thresholding (hysteresis):
   - Define two thresholds: low and high
   - Use the high threshold to start edge curves and the low threshold to continue them.
<table>
<thead>
<tr>
<th>Page</th>
<th>Canny edge detector (cont.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>39</td>
<td><strong>original image</strong></td>
</tr>
<tr>
<td>40</td>
<td><strong>Gradient magnitude</strong></td>
</tr>
</tbody>
</table>
Non-maximum suppression and hysteresis thresholding

We wish to mark points along the curve where the gradient magnitude is biggest.

We can do this by looking for a maximum along a slice normal to the curve (non-maximum suppression).

These points should form a curve.

There are then two algorithmic issues: at which point is the maximum, and where is the next one?
Non-maximum suppression (cont.)

At pixel q, we have a maximum if the value of the gradient magnitude is larger than the values at both p and at r.

Interpolation provides these values.

Edge linking

Assume the marked point 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 (here either r or s).
Hysteresis Thresholding

Reduces false edge pixels. It uses a low ($T_L$) and a high threshold ($T_H$) to create two additional images from the gradient magnitude image $g(x,y)$:

$$
g_L(x,y) = \begin{cases} g(x,y) & g(x,y) \geq T_L \\ 0 & \text{otherwise} \end{cases}, \quad g_H(x,y) = \begin{cases} g(x,y) & g(x,y) \geq T_H \\ 0 & \text{otherwise} \end{cases}
$$

$g_L(x,y)$ has more non zero pixels than $g_H(x,y)$. We eliminate from $g_L(x,y)$ all the common non zero pixels:

$$g_L(x,y) = g_L(x,y) - g_H(x,y)$$

$g_L(x,y)$ and $g_H(x,y)$ may be viewed as weak and strong edge pixels. Canny suggested a ratio of 2:1 to 3:1 between the thresholds.

Hysteresis Thresholding (cont.)

- After the thresholdings, all strong pixels are assumed to be valid edge pixels. Depending on the value of $T_H$, the edges in $g_H(x,y)$ typically have gaps.
- All pixels in $g_L(x,y)$ are considered valid edge pixels if they are 8-connected to a valid edge pixel in $g_H(x,y)$.
Hysteresis thresholding (cont.)

- High threshold (strong edges)
- Low threshold (weak edges)
- Hysteresis threshold

Source: L. Fei-Fei

Canny vs LoG

Image
Thresholded gradient

LoG
Canny

C. Nikou – Image Analysis (T-14)
Edge Linking

Even after hysteresis thresholding, the detected pixels do not completely characterize edges completely due to occlusions, non-uniform illumination and noise. Edge linking may be:

- **Local**: requiring knowledge of edge points in a small neighborhood.
- **Regional**: requiring knowledge of edge points on the boundary of a region.
- **Global**: the Hough transform, involving the entire edge image, will be presented at a later stage in the course.
Edge Linking by Local Processing

A simple algorithm:
1. Compute the gradient magnitude and angle arrays \( M(x,y) \) and \( a(x,y) \) of the input image \( f(x,y) \).
2. Let \( S_{xy} \) denote the neighborhood of pixel \((x,y)\).
3. A pixel \((s,t)\) in \( S_{xy} \) is linked to \((x,y)\) if:

\[
|M(x,y) - M(s,t)| \leq E \quad \text{and} \quad |a(x,y) - a(s,t)| \leq A
\]

A record of linked points must be kept as the center of \( S_{xy} \) moves.
Computationally expensive as all neighbors of every pixel should be examined.

Edge Linking by Local Processing (cont.)

A faster algorithm:
1. Compute the gradient magnitude and angle arrays \( M(x,y) \) and \( a(x,y) \) of the input image \( f(x,y) \).
2. Form a binary image:

\[
g(x,y) = \begin{cases} 
1 & M(x,y) \geq T_M \quad \text{and} \quad a(x,y) \in [A-T_A, A+T_A] \\
0 & \text{otherwise}
\end{cases}
\]

3. Scan the rows of \( g(x,y) \) (for \( A=0 \)) and fill (set to 1) all gaps (zeros) that do not exceed a specified length \( K \).
4. To detect gaps in any other direction \( A=\theta \), rotate \( g(x,y) \) by \( \theta \) and apply the horizontal scanning.
### Edge Linking by Local Processing (cont.)

<table>
<thead>
<tr>
<th>Image</th>
<th>Gradient magnitude</th>
<th>Horizontal linking</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="gradient1.png" alt="Gradient magnitude" /></td>
<td><img src="horizontal1.png" alt="Horizontal linking" /></td>
</tr>
<tr>
<td><img src="vertical1.png" alt="Vertical linking" /></td>
<td><img src="logicalOR1.png" alt="Logical OR" /></td>
<td><img src="morphological1.png" alt="Morphological thinning" /></td>
</tr>
</tbody>
</table>

We may detect the licence plate from the ratio width/length (2:1 in the USA).

---

### Edge Linking by Regional Processing

- Often, the location of regions of interest is known and pixel membership to regions is available.
- Approximation of the region boundary by fitting a polygon. Polygons are attractive because:
  - They capture the essential shape.
  - They keep the representation simple.
- Requirements
  - Two starting points must be specified (e.g. rightmost and leftmost points).
  - The points must be ordered (e.g. clockwise).
• Variations of the algorithm handle both open and closed curves.

• If this is not provided, it may be determined by distance criteria:
  – Uniform separation between points indicate a closed curve.
  – A relatively large distance between consecutive points with respect to the distances between other points indicate an open curve.

• We present here the basic mechanism for polygon fitting.

– Given the end points \(A\) and \(B\), compute the straight line \(AB\).

– Compute the perpendicular distance from all other points to this line.

– If this distance exceeds a threshold, the corresponding point \(C\) having the maximum distance from \(AB\) is declared a vertex.

– Compute lines \(AC\) and \(CB\) and continue.
Regional processing for edge linking is used in combination with other methods in a chain of processing.

![Regional Processing Diagram](image)

- Berkeley segmentation database:
  [http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/](http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/)
Learning to detect image boundaries

Learn from humans which combination of features is most indicative of a “good” contour?


What features are responsible for perceived edges?

Feature profiles (oriented energy, brightness, color, and texture gradients) along the patch’s horizontal diameter

What features are responsible for perceived edges? (cont.)


C. Nikou – Image Analysis (T-14)