Introduction to Computational and Biological Vision

CS 202-1-5261

Computer Science Department, BGU

Ohad Ben-Shahar
Drawing skills 101
Edge detection

Why look for edges? Physical causes for edges

Depth (boundary) discontinuities
Edge detection

Why look for edges? Physical causes for edges

Surface (normal) discontinuities
Edge detection

Why look for edges? Physical causes for edges
Edge detection

Why look for edges? Physical causes for edges
Edge detection

A (desired) edge detection mapping (labeling)

\[ E[I(x, y)]: \mathbb{R}^2 \rightarrow \Lambda = \{0, 1, 2, 3, 4, 5, \ldots\} \]

\[
E[I(x, y)] = \begin{cases} 
1 & \text{\( (x, y) \) is a boundary point} \\
2 & \text{\( (x, y) \) is a surface discontinuity point} \\
3 & \text{\( (x, y) \) is an illumination discontinuity point} \\
4 & \text{\( (x, y) \) is a reflectance discontinuity point} \\
\vdots \\
0 & \text{Otherwise (non edge point)}
\end{cases}
\]
Edge detection

Edge appearance in images
**Edge detection**

**Edges as curves**

**Definition:** A parametrized differentiable 2D curve is a differentiable map of an open interval \(A=(a,b)\) into \(\mathbb{R}^2\)

\[
\alpha : (a,b) \rightarrow \mathbb{R}^2 \\
\alpha(t) = (x(t), y(t))
\]

**Definition:** The set of points \(\alpha(A)\) is called the **trace** of the curve \(\alpha\).

**Definition:** Let \(\alpha(t)\) be a curve and let \(h(s)\) be a differentiable function on the open interval \(J\)

\[
h : J \rightarrow (a,b)
\]

Then the composite function

\[
\beta(s) = \alpha(h(s)) : J \rightarrow \mathbb{R}^2
\]

is a curve called \(\alpha\) **reparametrization** of \(\alpha\) by \(h\).
**Edge detection**

**Edges as curves**

**Definition:** The derivative vector

\[ \alpha'(t) = (x'(t), y'(t)) \]

is called the **velocity vector** of \( \alpha \). \( |\alpha'| \) is called the **speed** of the curve at \( t \).

\[ |\alpha'(t)| = \sqrt{(x'(t))^2 + (y'(t))^2} \]

**Definition:** The line \( \alpha(t) + s \cdot \alpha'(t) \) is called the **tangent line** of \( \alpha \) at point \( t \).

**Definition:** A parametrized differentiable 2D curve \( \alpha(t) \) is a said to be **regular** if

\[ \alpha'(t) \neq 0 \quad \forall t \in (a,b) \]
Edge detection

Edges as curves

\[ \alpha_1(t) = (t^2 - 4t, t^2 - 4) \]

\[ \alpha_2(t) = (t^3, t^3) \]

\[ \alpha_3(t) = (t^3, t^2) \]

\[ \alpha_4(t) = (t, |t|) \]
**Edge detection**

**Edges as curves**

Space curves: \( \alpha(t) = (x(t), y(t), z(t)) \)

Velocity vector: \( \alpha'(t) = (x'(t), y'(t), z'(t)) \)

Speed: \( |\alpha'(t)| = \sqrt{(x'(t))^2 + (y'(t))^2 + (z'(t))^2} \)

Tangent line: \( \alpha(t) + s \cdot \alpha'(t) \)

\( \alpha(t) = (a \cos(t), a \sin(t), bt) \)
**Edge detection**

**Edges as curves**

**Definition:** The **arclength** of a regular curve from the point \( t_0 \) is

\[
s(t) = \int_{t_0}^{t} |\alpha'(\tau)| \, d\tau
\]

**Definition:** A curve given in arclength parametrization is called **unit speed**, to have **arclength parametrization**, or to be **arclength parameterized**.

**Theorem:** If \( \alpha \) is a regular curve in \( \mathbb{R}^2 \), then there exists a reparametrization \( \beta \) of \( \alpha \) such that \( \beta \) has unit speed.
Edge detection

Edge appearance in images

\[ \alpha(t) \]

\[ I_{right}(x, y) \]

\[ I_{left}(x, y) \]
Edge detection

Edge appearance in images

- Step edge
- Roof edge
- Bright line
- Bright line
- Dark line
- Ramp edge
Edge detection

Step edge localization and detection – continuous signals

Edge detection heuristic #1
Given $I(x)$, its edge points occur at the (local) maxima of $|I'(x)|$
Edge detection

Step edge localization and detection – continuous signals

Edge detection heuristic #2
Given $I(x)$, its edge points occur at the zero crossing of $I''(x)$
**Edge detection**

Step edge localization and detection – discrete signals

\[
I'(x) \approx \frac{I(x+h) - I(x)}{h} + o(h)
\]

\[
I'(x) = \frac{I(x+h) - I(x-h)}{2h} + o(h^2)
\]

\[
I'(x) = \frac{I(x-2h) - 8I(x-h) + 8I(x+h) - I(x+2h)}{12h} + o(h^4)
\]

Edge detection via (central) differences operation is an instance of convolution!!
**Edge detection**

Step edge localization and detection – discrete signals

Edge detection via (central) differences operation is an instance of convolution!!
Edge detection

Step edge localization and detection – discrete signals

![Graph showing edge detection and step edge localization]

-1 0 1
Edge detection

Step edge localization and detection – discrete signals

![Graphs showing edge detection and its derivatives.](image-url)
Edge detection

Step edge localization and detection – discrete signals
Edge detection

Step edge localization and detection – discrete signals

$I(x)$

$I'(x)$
Edge detection

Step edge localization and detection – discrete signals
**Edge detection**

Gradient-based edge detection in images

\[ \nabla I = \left( \frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right) = \left( I_x, I_y \right) \]

Proposition:

The gradient vector points in the direction where the image changes “the most.”

**Edge detection heuristic #3**

Given \( I(x,y) \), its edge points occur at local (directional) maxima of \( |\nabla I| \).
**Edge detection**

Gradient-based edge detection in images

\[ |\nabla I| = L_2[\nabla I] = \sqrt{I_x^2 + I_y^2} \]

\[ |\nabla I| = L_1[\nabla I] = |I_x| + |I_y| \]

\[ I_x \approx \frac{I(x+1, y) - I(x-1, y)}{2} \]

\[ I_y \approx \frac{I(x, y+1) - I(x, y-1)}{2} \]
Edge detection

Gradient-based edge detection in images

\[ \theta(x, y) = \tan^{-1}\left( \frac{I_y}{I_x} \right) \]

\[ \theta^\perp(x, y) = \tan^{-1}\left( -\frac{I_x}{I_y} \right) \]
**Edge detection**

Laplacian-based edge detection in images

\[ \Delta I = \nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} \]

**Edge detection heuristic #4**

Given \( I(x,y) \), its edge points occur at the zero crossing of \( \Delta I \)
Edge detection

Laplacian-based edge detection in images

\[ \Delta I = \nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} \]

\[ \frac{\partial^2 I}{\partial x^2} \approx \frac{I(x-h, y) - 2I(x, y) + I(x+h, y)}{h^2} \]

\[ \frac{\partial^2 I}{\partial y^2} \approx \frac{I(x, y-h) - 2I(x, y) + I(x, y+h)}{h^2} \]

\[ \nabla^2 I \approx \frac{I(x-h, y) + I(x, y-h) - 4I(x, y) + I(x+h, y) + I(x, y+h)}{h^2} \]
Edge detection

Detecting 2D edge-like structures

$I(x, y)\quad |I_x|\quad |I_y|\quad |\nabla I|$
**Edge detection**

Detecting 2D edge-like structures

\[ I(x, y) \quad \nabla^2 I \quad ZC[\nabla^2 I] \]
Edge detection

Detecting 2D edge-like structures

\[ I(x, y) \quad \nabla^2 I \quad ZC[\nabla^2 I] \]
**Edge detection**

**Signal vs. Noise**

**Noise:**

Irrelevant or meaningless data that is mixed into the signal, usually due to imperfect performance of the measurement device.

**Basic noise models:**

- **Additive**
  \[
  \tilde{I}(x) = I(x) + \eta(x)
  \]

- **Salt & Pepper**
  \[
  \tilde{I}(x) = \begin{cases} 
  I(x) & \text{with probability } 1-p \\
  \min & \text{with probability } p/2 \\
  \max & \text{with probability } p/2 
  \end{cases}
  \]
Edge detection

Signal vs. Noise

\[ \sigma = 0.001 \]
\[ \sigma = 0.003 \]
\[ \sigma = 0.005 \]
\[ \sigma = 0.007 \]
\[ p = 0.05 \]
\[ p = 0.15 \]
\[ p = 0.25 \]
\[ p = 0.35 \]
Edge detection

Detecting 2D edge-like structures

$I(x, y)$  $|I_x|$  $|I_y|$  $|\nabla I|$
Edge detection

Detecting 2D edge-like structures

\[ I(x, y) \quad |I_x| \quad |I_y| \quad |\nabla I| \]
Edge detection

Detecting 2D edge-like structures

\[ I(x, y) \quad \nabla^2 I \quad ZC[\nabla^2 I] \]
Edge detection

Detecting 2D edge-like structures

\[ I(x, y) \quad \nabla^2 I \quad ZC[\nabla^2 I] \]
Edge detection

Detecting 2D edge-like structures

\[ I(x, y) \quad |I_x| \quad |I_y| \quad |\nabla I| \]
Edge detection

Detecting 2D edge-like structures

$I(x, y)$  \hspace{1cm} $|I_x|$  \hspace{1cm} $|I_y|$  \hspace{1cm} $|\nabla I|$
Edge detection

Detecting 2D edge-like structures

\begin{align*}
I(x, y) & \quad \nabla^2 I & \quad ZC[\nabla^2 I]
\end{align*}
Edge detection

Detecting 2D edge-like structures

$I(x, y)$  \hspace{1cm}  \nabla^2 I  \hspace{1cm}  ZC[\nabla^2 I]$
Edge detection

Detecting 2D edge-like structures

\[ I(x, y) \quad |\nabla I| \quad \text{ZC}[\nabla^2 I] \]
Edge detection

Detecting 2D edge-like structures

$I(x, y)$

$|\nabla I|$

$ZC[\nabla^2 I]$
Edge detection

Detecting 2D edge-like structures

\[ I(x, y) \quad |\nabla I| \quad ZC[\nabla^2 I] \]
Edge detection

Detecting 2D edge-like structures

$I(x, y)$  \hspace{1cm} $|\nabla I|$  \hspace{1cm} $ZC[\nabla^2 I]$
Edge detection

Detecting 2D edge-like structures

$I(x, y)$  \hspace{1cm}  $|\nabla I|$  \hspace{1cm}  $ZC[\nabla^2 I]$
**Edge detection**

Detecting 2D edge-like structures

\[ I(x, y) \]

\[ |\nabla I| \]

\[ ZC[\nabla^2 I] \]
Edge detection

Image denoising

Signal
changes slowly (most of the time)

Noise
changes fast, zero mean

What can reduce noise without affecting signal too much?

Smoothing
Averaging
Weighted sum
Convolution
Edge detection

Average-based image denoising
Edge detection

Average-based image denoising

Original

Window size 5

Window size 9

Window size 17
Edge detection

Detecting 2D edge-like structures

$I(x, y)$

$ZC[\nabla^2 I]$
Edge detection

Detecting 2D edge-like structures

$I(x, y)$
$|\nabla I|$
$ZC[\nabla^2 I]$
Edge detection

Detecting 2D edge-like structures

$I(x, y)$

$|\nabla I|$
Edge detection

Detecting 2D edge-like structures

$I(x, y)$

$|\nabla I|$

$ZC[\nabla^2 I]$
Edge detection

Combining smoothing and differentiation for edge detection

\[
(I * K_s) * K_d
\]

\[
I * (K_s * K_d)
\]
Combining smoothing and differentiation for edge detection

\[
(I \ast (K_{sd})) \ast K_d
\]
**Edge detection**

**Sobel edge detector**

\[
K_x = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} * \begin{bmatrix} -1 & +1 \\ -1 & +1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix}
\]

\[
K_y = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} * \begin{bmatrix} -1 & -1 \\ +1 & +1 \end{bmatrix} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix}
\]

\[
I_x = I * K_x \\
I_y = I * K_y \\
|\nabla I| = \sqrt{I_x^2 + I_y^2}
\]
Edge detection

Sobel edge detector – results on noisy images
Edge detection

Sobel edge detector – results on noisy images
Edge detection

Sobel edge detector – results on noisy images
Edge detection

Sobel edge detector – results on noisy images
**Edge detection**

**Gaussian-based smoothing**

\[
G_\sigma(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}
\]

\[
G_\sigma(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}
\]
**Edge detection**

**Gaussian-based smoothing**

Why Gaussian (as opposed to, say, simple window averaging)?

1. Contribution of neighbors weighted by distance
2. Isotropic (doesn’t prefer one direction over the other)
3. Symmetric about the origin.
4. Infinitely differentiable (as smooth as it gets)
5. Fourier transform of a Gaussian is another Gaussian (in the frequency domain)
6. Convolution of two Gaussians is a Gaussian \( G_{\sigma_1} \ast G_{\sigma_2} = G_{\sqrt{\sigma_1^2 + \sigma_2^2}} \)
7. Gaussians are separable \( G_{\sigma} (x, y) = G_{\sigma} (x) \cdot G_{\sigma} (y) \)
Edge detection

Gaussian-based smoothing

Why Gaussian (as opposed to, say, simple window averaging)?

8. N\textsuperscript{th} derivative of a Gaussian can be approximated by a appropriately substracting two (N-1)\textsuperscript{th} derivatives.

\[ G_{\sigma}(x) \quad G_{\sigma}'(x) \quad G_{\sigma}''(x) \]

\[ \Rightarrow \quad \Rightarrow \]

\[ \Rightarrow \quad \Rightarrow \]
Edge detection

Laplacian of Gaussian (LOG) and Lateral Inhibition
**Edge detection**

Marr-Hildreth edge detector (1980)

\[ MH(I) = R_{|\nabla I|} \left( ZC[\nabla^2 (I * G_\sigma)] \right) \]
Edge detection

Marr-Hildreth edge detector (1980)

\[ MH(I) = R_{|\nabla I|} \left( ZC\left[ (I * G_\sigma) * L \right] \right) \]
**Edge detection**

Marr-Hildreth edge detector (1980)

\[ MH(I) = R_{|\nabla I|}(ZC[I \ast (G_\sigma \ast L)]) \]
**Edge detection**

Marr-Hildreth edge detector (1980)

\[ MH(I) = R_{|\nabla I|}(ZC[I \ast (L \ast G_\sigma)]) \]
**Edge detection**

Marr-Hildreth edge detector (1980)

\[ MH(I) = R_{|\nabla I|}(ZC[I \ast LOG_{\sigma}]) \]
**Edge detection**

Marr-Hildreth edge detector (1980)

\[ MH(I) = R_{\vert \nabla I \vert} \left( ZC[I * LOG_{\sigma}] \right) \]

\[ LOG_{\sigma}(x, y) = a \left( 2 - \frac{x^2 + y^2}{\sigma^2} \right) e^{-\frac{(x^2 + y^2)}{2\sigma^2}} \]

The Mexican Hat Operator
Edge detection

Marr-Hildreth edge detector (1980)

\[ \ast G_{\sigma_1} \ast G_{\sigma_2} \ast G_{\sigma_{n-1}} \ast G_{\sigma_n} \]

...
**Edge detection**

Marr-Hildreth edge detector (1980)
Edge detection

Marr-Hildreth edge detector (1980)
Edge detection

Marr-Hildreth edge detector (1980)

Larger $\sigma$

Larger threshold
Edge detection

Marr-Hildreth edge detector (1980)
Edge detection

Marr-Hildreth edge detector (1980)

Larger σ

Larger threshold

[Diagram showing edge detection results with varying σ and threshold values]
Edge detection

Marr-Hildreth edge detector (1980)
Edge detection

Marr-Hildreth edge detector (1980)

Larger σ

Larger threshold
Edge detection

Marr-Hildreth edge detector (1980)
Edge detection

Marr-Hildreth edge detector (1980)
Edge detection

Marr-Hildreth edge detector (1980)
Edge detection

Marr-Hildreth edge detector (1980)
Edge detection

Marr-Hildreth edge detector (1980)
Edge detection

Desired: $E[I(x, y)]: \mathbb{R}^2 \rightarrow \Lambda = \{0, 1, 2, 3, 4, 5, \ldots\}$

Achieved:

Given these edge detection results, is edge detection solved?