Filters in Image Processing

Edge Detection

David Svoboda

email: svoboda@fi.muni.cz
Centre for Biomedical Image Analysis
Faculty of Informatics, Masaryk University, Brno, CZ

CBIA

November 15, 2019

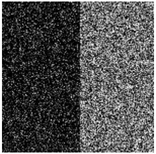
Outline

Motivation

Edge detection – the most commonly used operation in image analysis.

- black-white interface

- black-white interface with noise

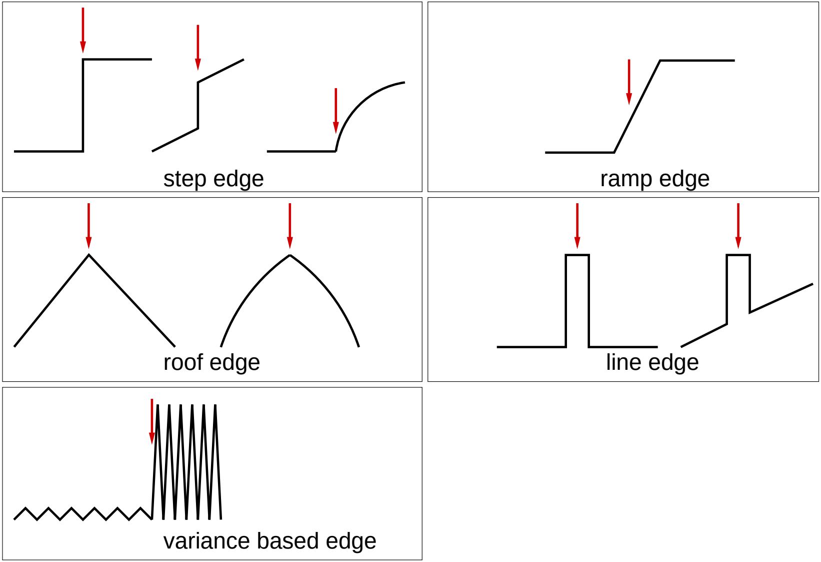


- texture interface

- colour interface

Motivation

What is an edge?



Motivation

What is an edge?

There is a number of possible definitions of an edge:

- step edge – the edge is simply a change in grey level occurring at one specific location
- ramp edge – the actual position of the edge is considered to be the center of the ramp
- roof edge – lambda shaped signal
- line edge – δ impulse in signal
- variance (texture) base edge – a change in variance levels

Notice: Edges are significant and abrupt changes in a signal.

- First derivative based
  - Gradient magnitude – strength of an edge:

$$|\nabla f(x,y)|, \quad \left|\frac{\partial f}{\partial x}\right| + \left|\frac{\partial f}{\partial y}\right|, \quad \text{or} \quad \max\left\{\left|\frac{\partial f}{\partial x}\right|, \left|\frac{\partial f}{\partial y}\right|\right\}$$

  - Gradient direction – direction perpendicular to an edge:

$$\nabla f(x,y) \quad \text{or} \quad \theta = \tan^{-1}\left(\frac{\frac{\partial f}{\partial y}}{\frac{\partial f}{\partial x}}\right)$$

- Second derivative based – zero crossings of the second derivative
- Template matching based

Example: High-frequent 1-D perturbation

$$f(x) = \varepsilon \sin\left(\frac{x}{\varepsilon^2}\right)$$

become arbitrary small for $\varepsilon \to 0$. However, its derivative

$$f'(x) = \frac{1}{\varepsilon} \cos\left(\frac{x}{\varepsilon^2}\right)$$

exceeds all bounds.

Notice: High-frequent fluctuations (noise) in the original signal can create unbounded perturbation in its derivatives.

First Derivative Based
Analysis

Interpretation in the Fourier domain:

- 1D:

$$\mathcal{F}\left(\frac{\partial^m f}{\partial x^m}\right)(\omega) = (2\pi i\omega)^m \mathcal{F}(f)(\omega)$$

- 2D:

$$\mathcal{F}\left(\frac{\partial^{m+n} f}{\partial x^n \partial y^m}\right)(\omega_x, \omega_y) = (2\pi i\omega_x)^n (2\pi i\omega_y)^m \mathcal{F}(f)(\omega_x, \omega_y)$$

Derivatives in the spatial domain lead to the multiplication in the Fourier domain. Thus, high-frequency components (e.g. noise) are amplified.

Remedy: Perform lowpass (e.g. Gaussian smoothing) filtering before computing derivative!

First Derivative Based

Gradient Estimator

$$|\nabla f(m,n)| = \sqrt{(\Delta_x f(m,n))^2 + (\Delta_y f(m,n))^2}$$

- Version 1:

$$\begin{aligned} \Delta_x f(m,n) &= f(m,n) - f(m-1,n) \\ \Delta_y f(m,n) &= f(m,n) - f(m,n-1) \end{aligned}$$

- Version 2:

$$\begin{aligned} \Delta_x f(m,n) &= f(m+1,n) - f(m-1,n) \\ \Delta_y f(m,n) &= f(m,n+1) - f(m,n-1) \end{aligned}$$

Notice: Δ . . . difference operator

- Diagonally oriented operator
- One of the oldest edge detectors with the following convolution masks:

+1	0
0	−1

(a) R_x

0	+1
−1	0

(b) R_y

Figure: Roberts kernels

$$|\nabla f(m,n)| = |f(m,n) - f(m+1,n+1)| + |f(m,n+1) - f(m+1,n)|$$

- Based on two convolution kernels S_x and S_y:

1	0	−1
2	0	−2
1	0	−1

(a) S_x

1	2	1
0	0	0
−1	−2	−1

(b) S_y

Figure: Sobel kernels

$$|\nabla f(m,n)| \quad = \quad \sqrt{(\Delta_x f_{y-smooth}(m,n))^2 + (\Delta_y f_{x-smooth}(m,n))^2}$$

John Canny [Canny-86] specified three **criteria** that an edge detector must address:

- Error rate – the edge detector should respond only to edge, and should find all of them; no edges should be missed
- Localization – the distance between the edge pixels as found by the edge detector and the actual edge should be as small as possible
- Response – the edge detector should not identify multiple edge pixels where only a single edge exists

Canny assumed:

- A step edge subject to white Gaussian noise.
- The edge detector was a convolution filter p that would smooth the noise and locate the edges.
- The problem was to identify the filter that optimizes the three edge detection criteria.

In one dimension, the response of the filter $p(x)$ of width W to an edge is given by the convolution:

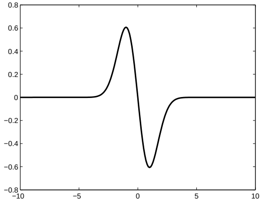
$$h(x) = \int_{-W}^{W} f(t)p(x-t)dt$$

where $f(t)$ denotes the input signal. The three criteria are expressed as:

Error rate:	Localization:	Response:
$SNR = \frac{A \int_{-W}^{0} p(x)dx}{\sigma \int_{-W}^{W} p^2(x)dx}$	$Loc = \frac{Ap'(0)}{\sigma \int_{-W}^{W} [p'(x)]^2 dx}$	$x_{zc} = \pi \left(\frac{\int_{-\infty}^{\infty} p^2(x)dx}{\int_{-\infty}^{\infty} p'^2(x)dx} \right)^{\frac{1}{2}}$

- Canny attempts to find the filter p that maximizes the product $SNR \times Loc$ subject to the multiple-response constraint.
- The result is too complex to be solved analytically.
- An efficient approximation turns out to be the first derivative of a Gaussian $g(x) = e^{-\frac{x^2}{2\sigma^2}}$:

$$p(x) \approx g'(x) = -\frac{x}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}}$$



- First, the gradient direction $r(x, y)$ is estimated at some point (x, y). If the image is noise free then

$$r(x, y) = \nabla f(x, y).$$

- Unfortunately, the image is usually noisy therefore we smooth the image by Gaussian

$$G(x, y) = e^{-(x^2+y^2)/2\sigma^2}$$

  Thus, we have

$$r(x, y) \approx \frac{\nabla (G * f)(x, y)}{\|\nabla (G * f)(x, y)\|}.$$

- We know that 1D Canny filter is equal to the derivative of the Gaussian. For that reason we compute

$$G_r = \frac{\partial G}{\partial r}$$

- Edge points show up as local maxima in the gradient image, and so if there is an edge passing through (x, y) in the direction $r(x, y)$ then there will be a local maximum in the image convolved with G_r, so that

$$\frac{\partial}{\partial r}(G_r * f) = 0.$$

- The gradient magnitude at this point will be:

$$\|G_r * f\| = \|(r \cdot \nabla G) * f\| = \|r\| \|\nabla G * f\|.$$

- Note that

$$(\nabla G * f) = \left(\frac{\partial G}{\partial x} * f, \frac{\partial G}{\partial y} * f \right)$$

and

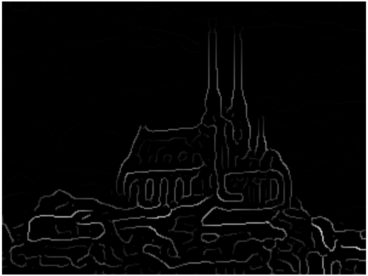
$$\frac{\partial G}{\partial x} = \frac{\partial}{\partial x} e^{-(x^2+y^2)/2\sigma^2} = -2xe^{-(x^2+y^2)/2\sigma^2} = g'(x)g(y)$$

1. Read in the image f to be processed.
2. Create a 1D Gaussian mask g to convolve with f. The standard deviation σ of this Gaussian is a parameter to the edge detector.
3. Create a 1D mask for the first derivative of the Gaussian in the x and y direction; call these g_x and g_y. The same σ value is used as in step 2 above.
4. Convolve the image f with g along the rows to give the x component image f_x, and down the columns to give the y component image f_y.
5. Convolve f_x with g_y (*orthogonal directions*) to give f'_x, the x component of f convolved with the derivative of the Gaussian, and convolve f_y with g_x to give f'_y.
6. The magnitude of the result is computed at each pixel (x, y) as:

$$|\nabla G * f(x, y)| = \sqrt{f'_x(x, y)^2 + f'_y(x, y)^2}$$

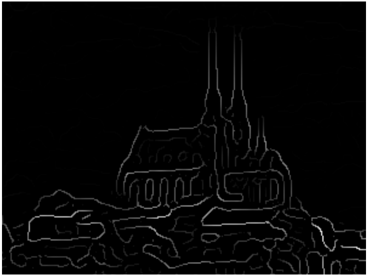
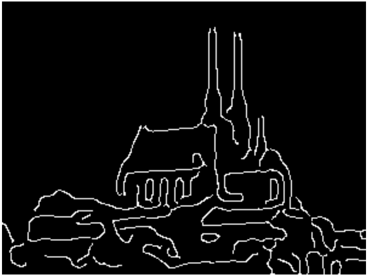
Nonmaxima Suppression

- Goal: thinning of edges to a width of 1 pixel
- In every edge pixel, consider the grid direction (out of 4 directions) that is "most orthogonal" to the edge.
- If one of the two neighbours in this direction has a larger gradient magnitude, remove the central pixel from the edge map.



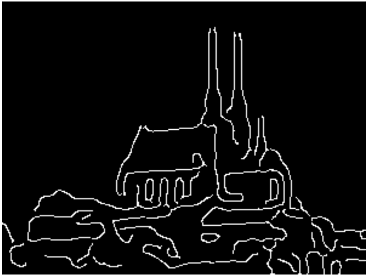
Hysteresis Thresholding

- Goal: extract only relevant edges.
- Use points above the upper threshold as seed points of relevant edges.
- Add all neighbours that are below the upper threshold, but above the lower threshold.

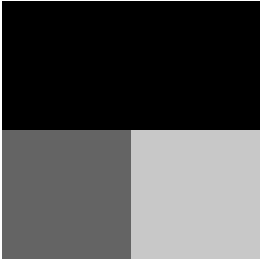



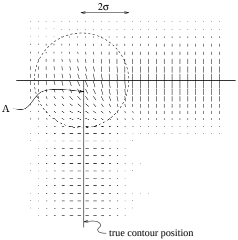
Some Important Properties

- One of the most popular edge detectors (benchmark)
- Taken as “ground truth” among the others
- Optimal under certain conditions (step edges & white Gaussian noise)
- Canny does not produce continuous edges



Uses the idea of Canny but modifies the "nonmaxima suppression" step, since the edge direction is not correct near corners and junctions:





- topological based approach
- thinning (nonmaxima suppression) is modified to preserve topological properties of the objects in the image

An example

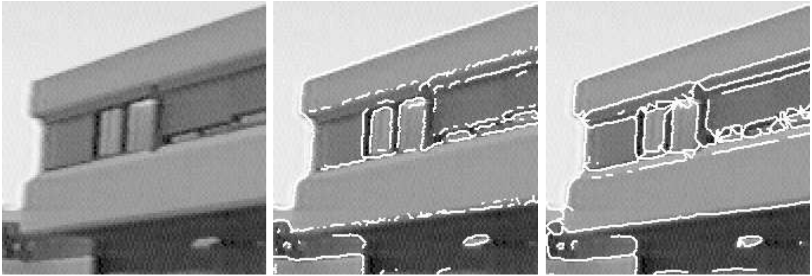


Figure: (left) original image; (centre) Canny output; (right) Rothwell modified edge detector output

Second Derivative Based Edge Detectors

Idea: A maximum of the first derivative, i.e. an edge, will occur at a zero crossing of the second derivative.

The most typical (1D) approximation:

$$\Delta^2 f(m) = \frac{f(m+1) - 2f(m) + f(m-1)}{h^2} + O(h^2)$$

Standard (2D) approximation using Laplacian:

$$\nabla^2 f(m,n) = f_{xx}(m,n) + f_{yy}(m,n)$$

$$\nabla^2 \approx \frac{1}{h^2} \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

Second Derivative Based Edge Detectors

Disadvantages:

- does not only detect maxima of the first derivative, but also the minima
- very sensitive to noise
- strong Gaussian smoothing is required $\rightarrow$ delocalization
- does not detect edge direction $\rightarrow$ first derivative evaluation is required

Advantages:

- generate closed contours
- rotationally symmetric
- orientation-independent (if the local intensity change is nearly linear)
- no input parameters but the width of Gaussian

Second Derivative Based Edge Detectors

Laplacian of Gaussian (Marr-Hildreth)

- Given smoothing kernel: $G(x,y) = -e^{-\frac{x^2+y^2}{2\sigma^2}}$
- Laplacian of $G(x,y)$:

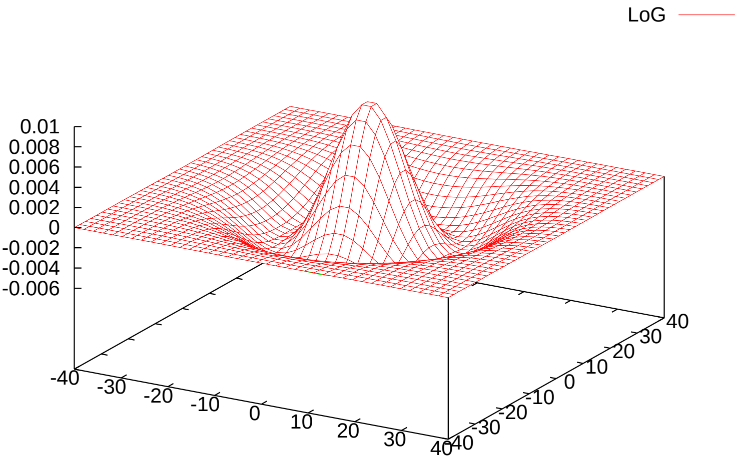
$$\nabla^2 G(x,y) = -\left[\frac{(x^2+y^2)-\sigma^2}{\sigma^4}\right] e^{-\frac{x^2+y^2}{2\sigma^2}}$$

is called "Laplacian of Gaussian (LoG)".

Example: 5×5 LoG filter mask

0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

Due to its shape, LoG is called the Mexican hat function:



Second Derivative Based Edge Detectors

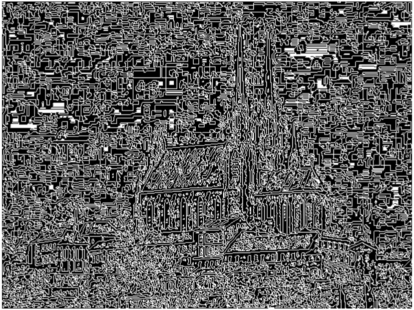
Laplacian of Gaussian – an example

$\sigma = 1.0$

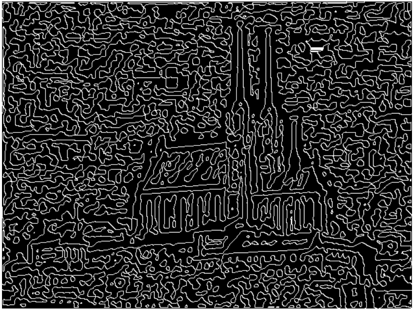
$\sigma = 3.0$

Second Derivative Based Edge Detectors

Laplacian of Gaussian – an example



$\sigma = 1.0$



$\sigma = 3.0$

- DoG is close approximation to the LoG filter
- Convolution kernel is given by

$$DoG = G_{\sigma_1} - G_{\sigma_2}$$

where $\sigma_1 < \sigma_2$

- Marr and Hildreth found out that ratio

$$\frac{\sigma_2}{\sigma_1} = 1.6$$

provides a good approximation to the LoG.

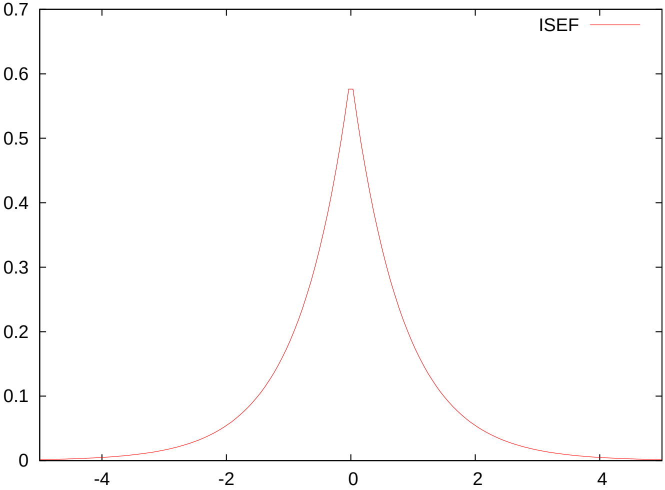
Shen and Castan designed infinite symmetric exponential filter (ISEF):

- alternative solution to Canny optimal edge detector
- they suggest minimizing (in 1D):

$$C_N^2 = \frac{4 \int_0^\infty g^2(x)dx \cdot \int_0^\infty g'^2(x)dx}{g^4(0)}$$

- the function that minimizes C_N is the optimal smoothing filter for an edge detector
- optimal filter function (ISEF for 1D): $g(x) = \frac{p}{2} e^{-p|x|}, \quad p > 0$
- optimal filter function (ISEF for 2D): $g(x, y) = a \cdot e^{-p(|x|+|y|)}$
- this produces better signal to noise ratios and better localization than Canny.

Shape of ISEF for $p = 1.2$:



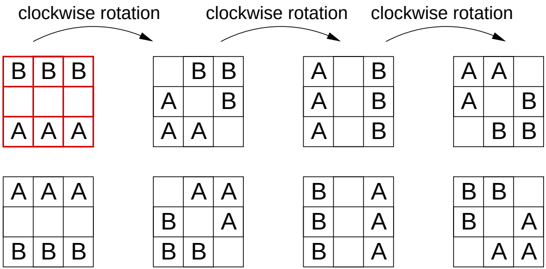
1. Convolve the input image with the ISEF
2. Localize edges by subtracting the original image from the smoothed one (similar to the Marr-Hildreth algorithm)
3. A binary Laplacian image is generated by setting all the positive valued pixels to 1 and all others to 0
4. The candidate pixels are on the boundaries of the regions in the binary image
5. Postprocessing:
    - false zero-crossing suppression (similar to Canny nonmaxima suppression)
    - hysteresis thresholding

Template Based Edge Detectors

The idea is to use a small discrete template as model of an edge.

2D specific:

- several convolution kernels are created by rotating one "seed kernel"
- kernel with maximum response (e.g. correlation) defines the result at given location



Kirsch operator:

–3	–3	5
–3	0	5
–3	–3	5

Robinson operator:

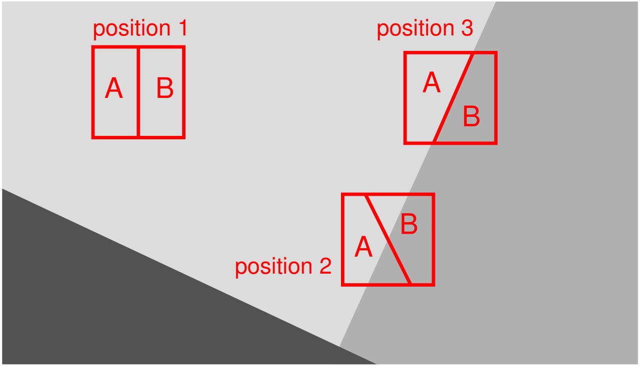
1	1	1
1	–2	1
–1	–1	–1

Prewitt operator:

1	1	1
0	0	0
–1	–1	–1

Sobel operator:

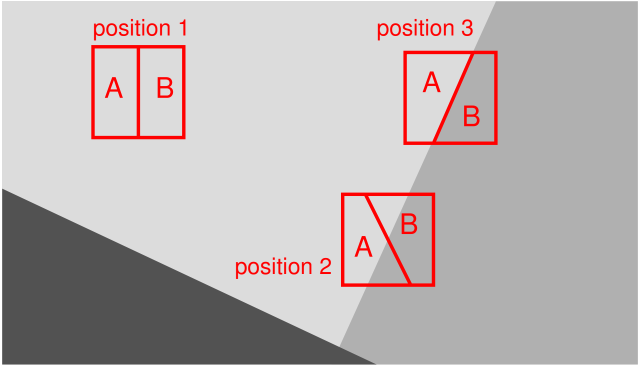
1	0	–1
2	0	–2
1	0	–1



- *position 1*: $mean_A = mean_B$
- *position 2*: $mean_A \approx mean_B$
- *position 3*: $|mean_A - mean_B|$ is very high number

Template Based Edge Detectors

(Nonlinear) Goodness-Of-Fit Test Based Edge Detection



- *position 1*: no match with given template
- *position 2*: bad match
- *position 3*: edge found

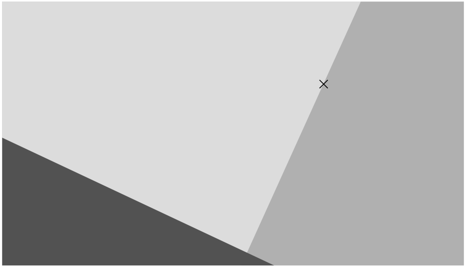
Basic principle

For each point (x, y) of the image f:

1. apply the mask:
2. measure & rotate
3. measure & rotate

   ...

4. measure
5. find the highest measure
6. $|\nabla f(x, y)|$ is the edge strength
7. α is the edge direction



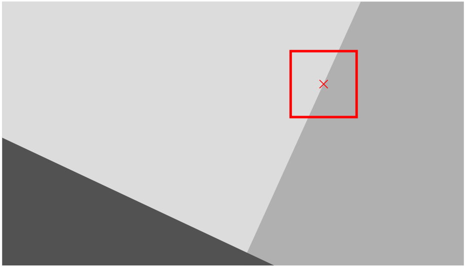
Basic principle

For each point (x, y) of the image f:

1. apply the mask:
2. measure & rotate
3. measure & rotate

   ...

4. measure
5. find the highest measure
6. $|\nabla f(x, y)|$ is the edge strength
7. α is the edge direction



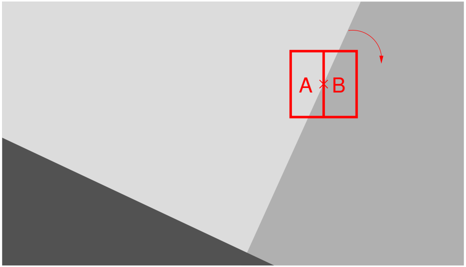
Basic principle

For each point (x, y) of the image f:

1. apply the mask:
2. measure & rotate
3. measure & rotate

   ...

4. measure
5. find the highest measure
6. $|\nabla f(x, y)|$ is the edge strength
7. α is the edge direction



- collect A pixels $\rightarrow \mathcal{A}$
- collect B pixels $\rightarrow \mathcal{B}$
- compute "goodness-of-fit" test over $\mathcal{A}$ and $\mathcal{B}$

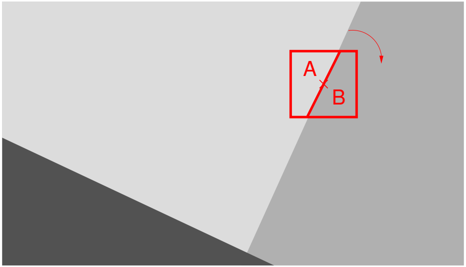
Basic principle

For each point (x, y) of the image f:

1. apply the mask:
2. measure & rotate
3. measure & rotate

   ...

4. measure
5. find the highest measure
6. $|\nabla f(x, y)|$ is the edge strength
7. α is the edge direction



- collect A pixels $\rightarrow \mathcal{A}$
- collect B pixels $\rightarrow \mathcal{B}$
- compute “goodness-of-fit” test over $\mathcal{A}$ and $\mathcal{B}$

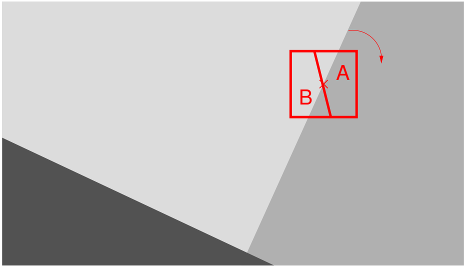
Basic principle

For each point (x, y) of the image f:

1. apply the mask:
2. measure & rotate
3. measure & rotate

   ...

4. measure
5. find the highest measure
6. $|\nabla f(x, y)|$ is the edge strength
7. α is the edge direction



- collect A pixels $\rightarrow \mathcal{A}$
- collect B pixels $\rightarrow \mathcal{B}$
- compute "goodness-of-fit" test over $\mathcal{A}$ and $\mathcal{B}$

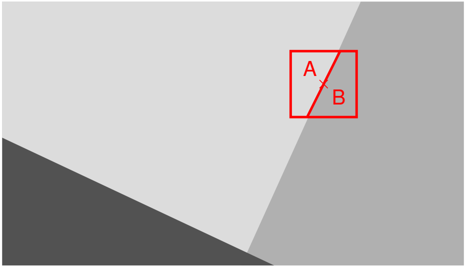
Basic principle

For each point (x, y) of the image f:

1. apply the mask:
2. measure & rotate
3. measure & rotate

   ...

4. measure
5. find the highest measure
6. $|\nabla f(x, y)|$ is the edge strength
7. α is the edge direction



$$|\nabla f(x, y)| = \max_{\alpha} \left(measure \left(\mathcal{A}, \mathcal{B} \right) \right)$$

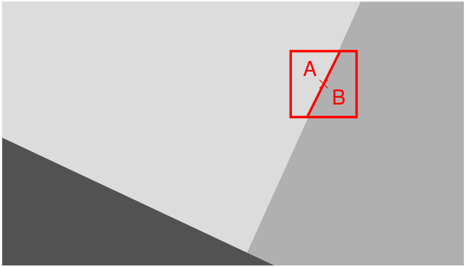
Basic principle

For each point (x, y) of the image f:

1. apply the mask:
2. measure & rotate
3. measure & rotate

   ...

4. measure
5. find the highest measure
6. $|\nabla f(x, y)|$ is the edge strength
7. α is the edge direction



$$|\nabla f(x, y)| = \max_{\alpha} \left(measure \left(\mathcal{A}, \mathcal{B} \right) \right)$$

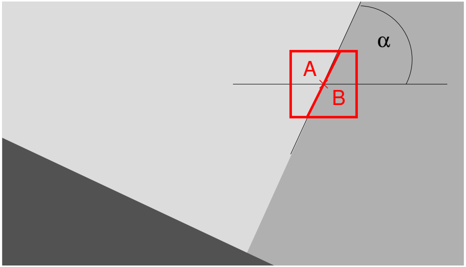
Basic principle

For each point (x, y) of the image f:

1. apply the mask:
2. measure & rotate
3. measure & rotate

   ...

4. measure
5. find the highest measure
6. $|\nabla f(x, y)|$ is the edge strength
7. α is the edge direction



$$|\nabla f(x, y)| = \max_{\alpha} \left(\mathit{measure} \left(\mathcal{A}, \mathcal{B} \right) \right)$$

The use of various statistics

The measure is a tool for edge detection in the location between two different neighbouring areas.

Two sample goodness-of-fit test deciding whether chosen datasets $\mathcal{A}$ and $\mathcal{B}$ differ may be:

- Student's T test (mean and variance)
- Fisher/Likelihood-test (variance)
- χ^2-test (frequency)
- Kolmogorov-Smirnov test (cumulative distribution)
- Wilcoxon test (distribution)
- simply mean difference
- etc ...

Advantages

- offer similar ability as traditional gradient based detectors
- give better performance on noisy images and texture images
- the statistical filter incorporates a process of edge tracking inherent within the algorithm

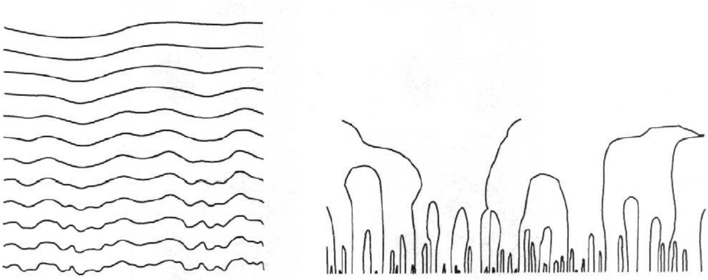
Drawbacks

- slower
- due to predefined templates these cannot find corners correctly

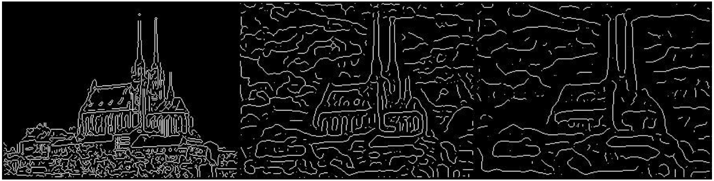
Scale-Space Based

Interesting Observation

- Structures that can be detected at a coarse scale σ can be traced back to smaller scales in order to improve their localization
- This has led to the notion of scale-space: Embed an image in a continuum of more and more smoother versions of it.



Scale-Space Based



(a) $\sigma = 1$ (b) $\sigma = 3$ (c) $\sigma = 5$

(d) $\sigma = 10$ (e) $\sigma = 20$ (f) $\sigma = 30$

Edge Evaluation Methods

Fundamentals

How to solve the problem of evaluating the performance of edge detectors?

Given ground truth (GT) image and the edge map (EM), we can report the following statistics:

- true positive (TP)
- false positive (FP)
- true negative (TN)
- false negative (FN)

Monitoring of only one measure may lead to wrong conclusions. Tuning a detector to increase the TP score generally also results in a higher FP score.

- sensitivity = TP/(TP+FN)
- specificity = TN/(TN+FP)
- accuracy = (TP+TN)/(TP+FN+TN+FP)

Edge Evaluation Method

List of the most utilized evaluation methods:

- Utilization of Canny's edge detector as a benchmark [Canny]
- Pratt's Figure of Merit [Pratt]
- Local Coherence [Kitchen]
- ROC curves [Bowyer]
- Pixel Correspondence Metric [Prieto]

Notice: Keep in mind, that the majority of similarity metrics manage only binary data (binary edges).

Pratt's Figure Of Merit (FOM)

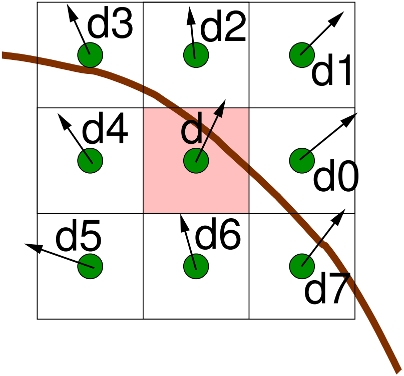
The similarity measure designed by Pratt is defined as follows:

$$FOM = \frac{1}{I_N} \sum_{i=1}^{I_{EM}} \frac{1}{1 + \beta d^2}$$

where

- I_{GT} ... number of pixels in GT
- I_{EM} ... number of pixels in edge map
- $I_N = \max(I_{GT}, I_{EM})$
- β ... scaling (magic) constant (typically set to $1/9$)
- d ... separation distance between an actual edge pixel in EM and its correct position in the GT
- $FOM = 1$ is valid for perfect match

Local Edge Coherence



The aim of this measure is to inspect the *thinness* and *coherence* of an edge in each pixel.

Legend:

- brown . . . edge
- pink . . . inspected pixel
- arrows . . . gradient direction

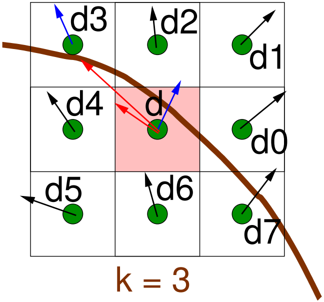
Given an edge direction d at some given $\mathbf{x}$ pixel we measure:

1. how well an edge pixel $\mathbf{x}$ is continued on the **left**

$$L(k) = \begin{cases} dist(d, d_k)\, dist\left(\frac{k\pi}{4}, d + \frac{\pi}{2}\right), & \text{if neighbour } k \text{ is an edge pixel} \\ 0, & \text{otherwise} \end{cases}$$

where d is the edge direction at the pixel being tested, d_0 is the edge direction at its neighbor to the right, d_1 is the direction of the upper-right neighbour, and so on counterclockwise about the pixel involved. The function *dist* is a measure of the angular difference between any two angles:

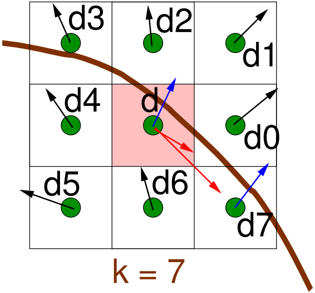
$$dist(\alpha, \beta) = \frac{\pi - |\alpha - \beta|}{\pi}$$



Local Edge Coherence

2. A similar function measures directional continuity on the **right** of the pixel **x**:

$$R(k) = \begin{cases} dist(d, d_k)\,dist\left(\frac{k\pi}{4}, d - \frac{\pi}{2}\right), & \text{if neighbour } k \text{ is an edge pixel} \\ 0, & \text{otherwise} \end{cases}$$



Local Edge Coherence

3. The overall continuity measure C is taken to be the average of the largest value of $L(k)$ and the largest value of $R(k)$.
4. An edge should be thin line, one pixel wide. The thinness measure T is the fraction of the six pixels in the 3×3 neighbourhood, excluding the center and the two pixels found by $L(k)$ and $R(k)$, that are the edge pixels.
5. The overall evaluation of the edge detector is

$$E_2 = \gamma C + (1 - \gamma) T$$

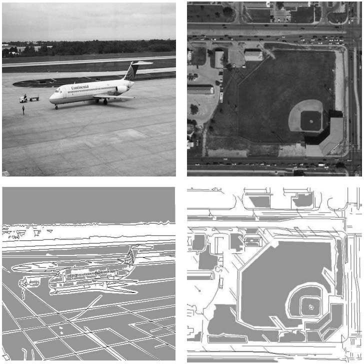
where γ is a constant.

ROC curves

Definition

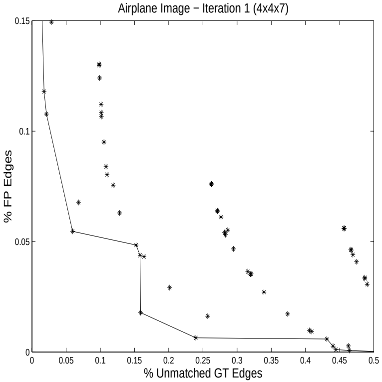
Let ground truth (GT) be a reference image in which:

- black . . . edge
- gray . . . no-edge
- white . . . “don’t care”



ROC = Receiver Operating Characteristics

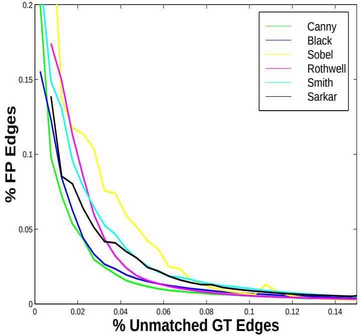
1. Sample the parameter space of an edge detector.
2. For each sample do
   - execute the edge detector,
   - evaluate FN and FP count,
   - put (FN, FP)-point into the graph.
3. Analyze the points and construct the ROC curve.



ROC curve construction

A point P appears on the ROC curve only if no other point is included in the axis-oriented rectangle demarcated by origin (0,0) and P.

- Average several ROC curves, each generated from different image.
- The ideal point is (FN,FP)=(0,0) → the ROC curve with the lower "area under the curve" is the better one.



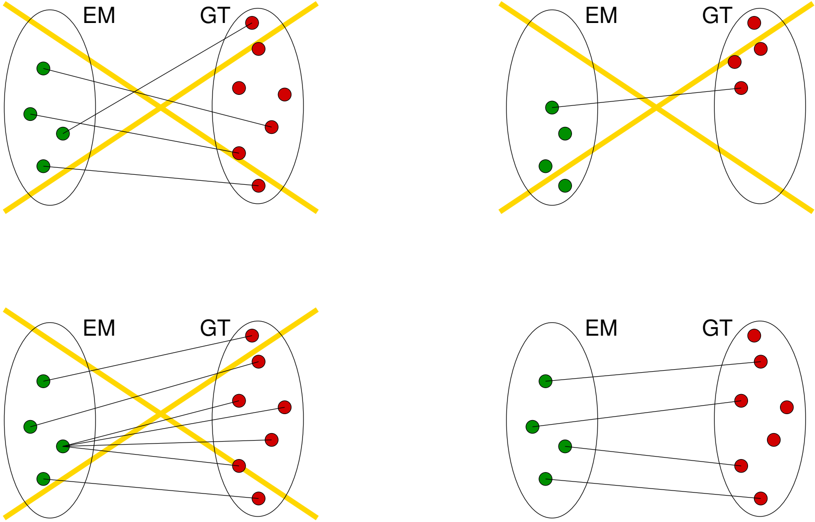
The common evaluation methods classify this situation

$$GT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad EM = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

as one correctly detected edge pixel, one misdetection, and one false alarm.

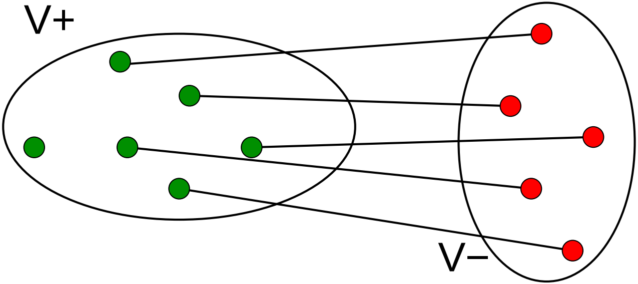
The only mistake is the small diagonal shift!

Pixel Correspondence Metric (PCM)



Let $\mathcal{G}(V, E)$ be an undirected graph with vertex set V and edge set E. The graph $\mathcal{G}$ is bipartite if the vertex set V can be partitioned into two disjoint sets V^+ and V^-:

- match $\mathcal{M}$ is a subset of E such that no two edges share a vertex
- vertex is matched if it is incident to an edge in $\mathcal{M}$ and unmatched otherwise
- edge is matched if it contained in $\mathcal{M}$ and unmatched otherwise



Pixel Correspondence Metric (PCM)

Separation

Let f and g be two images of the same dimensions. The separation S between the pixels in positions (i, j) and (k, l) is defined as:

$$S((i,j),(k,l)) = E(\max(|k-i|,|l-j|)),$$

where $E(d) \in [0;1]$ is a normalized function that represents a weighting dependent on the chessboard distance between pixels:

$$E(d) = (1, 0.9, 0.65, 0.5) \,|\, d = 0 \ldots 3$$

Pixel Correspondence Metric (PCM)

Cost of match

Let $\mathcal{M}(f,g)$ be some match between two images f and g. The cost of a match of two particular pixels $f(i,j)$ and $g(k,l)$ is:

$$C(f(i,j),g(k,l)) = 1 - S((i,j),(k,l))\left(1 - \frac{|f(i,j)-g(k,l)|}{\text{max value}}\right)$$

Example: The cost of match between two pixels $f(3,35)=140$ and $g(5,36)=130$ from 8-bit images is:

$$\begin{aligned} C(f(3,35),g(5,36)) &= 1 - S((3,35),(5,36))\cdot\left(1-\frac{|140-130|}{255}\right) \\ &= 1 - E(2)\left(1-\frac{10}{255}\right) \\ &= 1 - 0.69(0.961) \\ &= 0.337 \end{aligned}$$

Pixel Correspondence Metric (PCM)

Getting optimal match

- EM & GT ... two disjoint parts (V^+ and V^-) of bipartite graph.
- The weight of edge connecting pixels $f(i,j)$ and $g(k,l)$:

$$W = \lceil C(f(i,j), g(k,l)) \cdot (\text{max value}) \rceil$$

- The cost of match for the whole graph $C(\mathcal{M})$ is the accumulated value of
  - all the weights of the edges in $\mathcal{M}$ plus
  - the accumulated value of all the unmatched vertices (the value of the pixel that the vertex represents)
- Optimal match $\mathcal{M}_{opt}(f,g)$ is a match with minimal cost among all possible matches.

Pixel Correspondence Metric (PCM)

Definition

Pixel Correspondence Metric

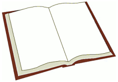
$$PCM_\eta(f,g) = 100 \left(1 - \frac{C(\mathcal{M}_{opt}(f,g))}{|f \cup g|}\right)$$

Some properties:

- $PCM_\eta(f,g) \in [0;100]$
- If images $f = g$ then $PCM_\eta(f,g) = 100$
- Search for optimal match in bipartite graphs is too hard. The common way is to solve the task locally.
- This method is capable of working with grayscale data.

Bibliography

- Lim, D. H. Robust edge detection in noisy images. Comput. Statist. Data Anal. v50. 803-812, 2006
- deSouza, P. Edge detection using sliding statistical tests. Comput. Vision Graphics Image Process. v23 i1. 1-14, 1983
- Bowyer K. W., Kranenburg Ch., Dougherty S. Edge detector evaluation using empirical ROC curves, Computer Vision and Image Understanding 84 (1), October 2001, 77-103
- Prieto M. S., Allen A. R. A similarity metric for edge images, IEEE TPAMI 25 (10): pp 1265-1273; 2003
- Kitchen, L. and Rosenfeld, A. Edge Evaluation Using Local Edge Coherence, IEEE TSMC, Vol 11. 9:597-605, 1981
- Canny, J. A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. 8, 6 (Nov. 1986), 679-698.
- Rothwell, C., Mundy, J., Hoffman, W. and Nguyen, V.-D. Driving Vision by Topology, Technical Report No. 2444, INRIA, 1994
- Marr, D. and Hildreth, E. Theory of edge detection. In Proc. Royal Soc. Lond., volume B 207, pages 187–217, 1980
- Shen, J. and Castan, S. An optimal linear operator for step edge detection. Computer Vision, Graphics and Image Processing, 54(2):112–133, March 1992
- Witkin, A. P. Scale-space filtering, Proc. 8th Int. Joint Conf. Art. Intell., Karlsruhe, Germany, 1019–1022, 1983



You should know the answers . . .

- What are the *pros and cons* of the first derivative based edge detection? Explain the idividual items.
- Compare Sobel and Canny's operator.
- Propose a simple pseudocode for *nonmaxima suppression* algorithm.
- In terms of edge detection, what does *zero-crossing* mean?
- How do we get/compute an edge direction by template based edge detectors?
- Explain the use of *sensitivity*, *specificity*, and *accuracy*. Show the examples.
- How would you measure the edge coherence? Explain in detail.
- When constructing the ROC curves, what is the size of parametric space for Roberts operator?
- When measuring the quality of edge detection, how would you assign the corresponding pixels from EM and GT?