Edges (= gradients), edge elements, and image sharpening

Václav Hlaváč

Czech Technical University in Prague
Czech Institute of Informatics, Robotics and Cybernetics
160 00 Prague 6, Jugoslávských partyzánů 1580/3, Czech Republic
http://people.ciirc.cvut.cz/hlavac, vaclav.hlavac@cvut.cz
also Center for Machine Perception, http://cmp.felk.cvut.cz

Thanks to O. Drbohlav, T. Svoboda and T. Werner for several slides.

Outline of the lecture:

- Motivation, edges, edgels.
- Three groups of edge operators.
- Finding edges by a convolution.
- Marr-Hildreth edge detector.
- Scale space.
- Canny edge detector.

Image preprocessing, the intro

The input is an image, the output is an image too.

The image is not interpreted.

The aim

- To suppress the **distortion** (e.g., correction of the geometric distortion caused by spherical shape of the Earth taken from a satellite).
- **Contrast** enhancement (which is useful only if the human looks at the image).
- **Noise** suppression.
- **Enhancement of some image features** needed for further image processing, e.g., edge elements finding.

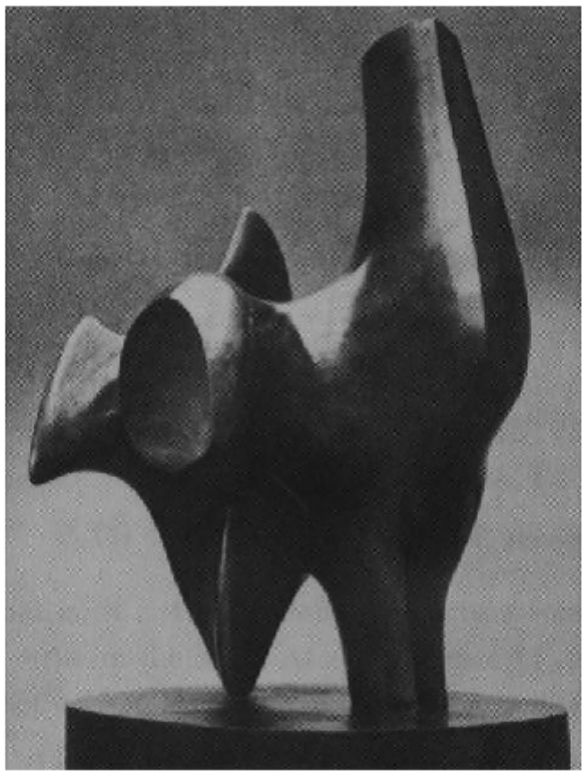
Motivation. What are the edges and edgels good for?

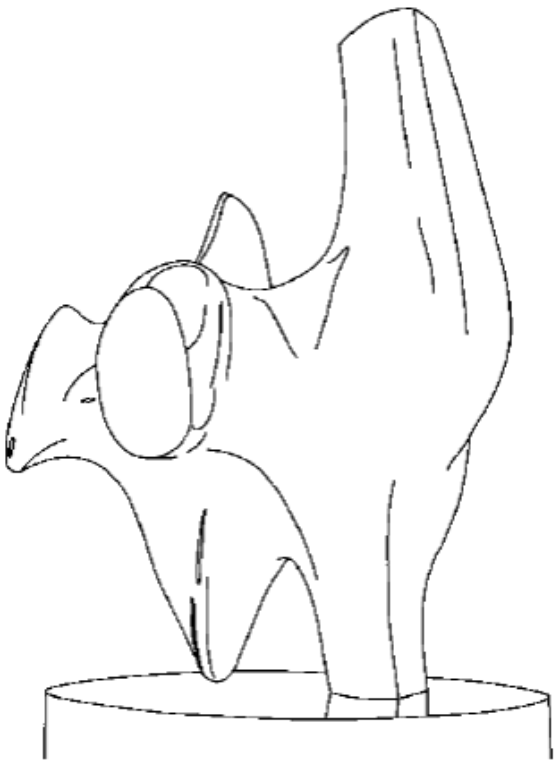
- Neurophysiological and psychophysical studies hint that locations in images, in which the image intensity changes sharply, i.e., at significant edges = edge elements (abbreviated edgels), are important for visual perception.
- These locations bear more useful knowledge than other locations in the image.
- Edgels often exhibit fair invariance to changes in illumination and/or viewpoint.
- Our **intention is**
  - either to **enhance**, i.e., strengthen, high frequencies using the edge sharpening operation
  - or **detect** $\Rightarrow$ significant edges.
- Edge (= gradient) computation/edgel detection is often used in computer vision for: recognition of image content, 3D reconstruction of the scene, solving correspondence problem, object tracking, etc.

Example – a drawing matches to edgels

Pablo Picasso, La Sieste 1919

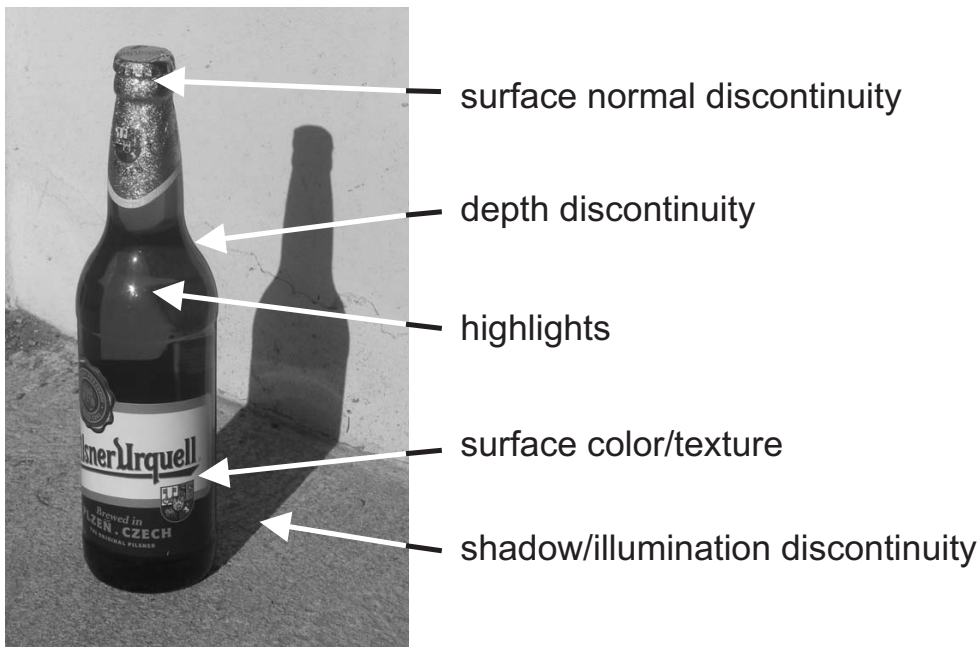
Example – edgels automatic detection





The origin of edge elements

Edgels are the result of discontinuities in the surface normal, in depth or in reflectance; they also appear due to highligts or irregularities in illumination (shadows).



Edge, edgel

Edge (= gradient)

- is a property of a point (pixel in discrete image) and its neighborhood;
- expresses the speed of the image function $f(x, y)$ (intensity) change and the direction of the maximal growth of it; i.e. it is the gradient $\nabla f(x, y)$;
- is a suitable discrete approximation of a gradient $\nabla f(x, y)$, i.e. is a two component vector.

Edge element (edgel)

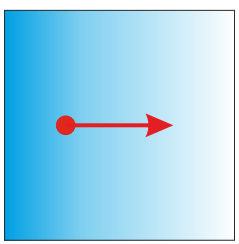
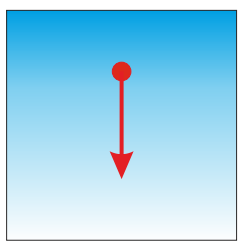
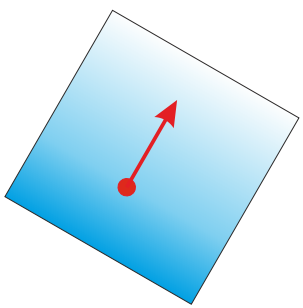
- is a point (pixel) with a significant gradient module.
- It implies that some points (pixels) in the image are egels and some are not edgels.
- Edgels are well localized in the image.
- Edgels positions are stable with respect to changing views.

Three families of edge computation/edgels detectors

Gradient calculation and edgel detectors based on:

1. Approximating maxima of first derivatives (Roberts, Prewitt, Sobel etc, Canny);
2. Finding zero-crossing of second derivatives (Marr-Hildreth);
3. Local approximation of image function by parametric model, usually a polynomial in x, y. Computing the derivative from the model parameters analytically. (Pioneered in computer vision by R. Haralick)

Gradient of the image function

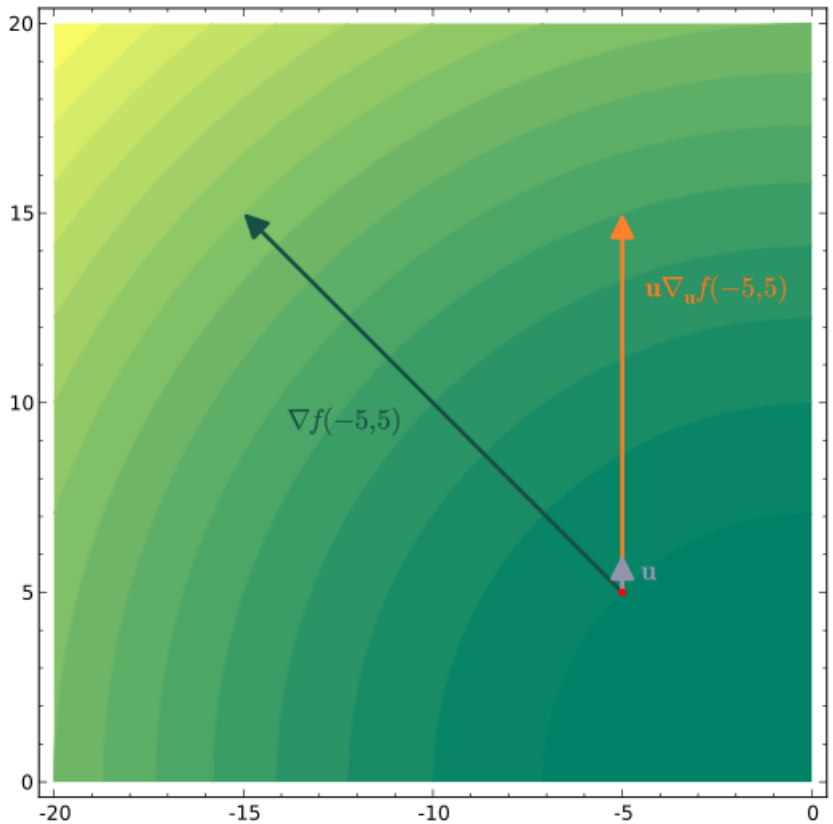
- For n variables in general, the gradient of a smooth function f is a vector of partial derivatives: $\nabla f(x_1, \ldots, x_n) = \left(\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n}\right)$
- For $n = 1$ (1D signal), the gradient simplifies to a (standard) derivative expressed in two notations: f' (Lagrange notation) or $\frac{\mathrm{d}f}{\mathrm{d}x}$ (Leibnitz notation).
- For $n = 2$ (2D signal), $\nabla f(x, y) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$
  -   

    $\nabla f = \left(\frac{\partial f}{\partial x}, 0\right)$ $\nabla f = \left(0, \frac{\partial f}{\partial y}\right)$ $\nabla f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$
  - $\nabla f(x, y)$ is often expressed in polar coordinates by its magnitude and angular direction ψ

    $\|\nabla f(x, y)\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$, $\psi = \arctan\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$.

Directional derivative

- Consider a 2D function $f(x, y)$, one specific point (x_0, y_0) and a direction given by a vector $\mathbf{u} = (u_1, u_2)$. The corresponding unit vector $\hat{\mathbf{u}}$ to vector $\mathbf{u}$ is $\hat{\mathbf{u}} = \frac{\mathbf{u}}{|\mathbf{u}|}$.
- The directional derivative $\nabla_{\mathrm{u}} f(x_0, y_0)$ is the rate the function $f(x, y)$ changes at a point (x_0, y_0) in the direction $\mathbf{u}$. It is defined in two alternative ways
  - $\nabla_{\mathrm{u}} f(x_0, y_0) \equiv \nabla f(x, y) \frac{\mathbf{u}}{|\mathbf{u}|}$ or
  - $\frac{\mathrm{d}f(x,y)}{\mathrm{d}u} \equiv \hat{\mathbf{u}} \cdot \nabla f(x, y) = u_1 \frac{\partial f(x,y)}{\partial x} + u_2 \frac{\partial f(x,y)}{\partial y}$.



Discrete approximation of a gradient

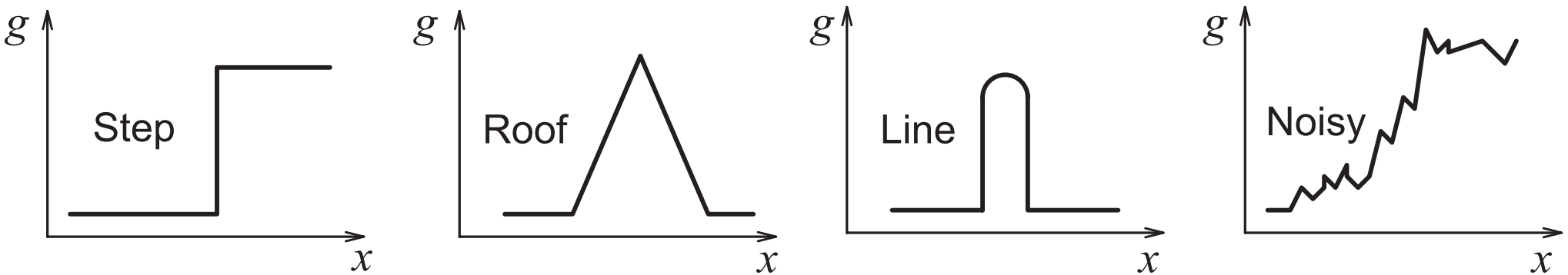
Can be done by either of these two ways (with similar results).

- Reconstruct the continuous function from the discrete one and compute its derivative.
- Approximate the derivative by finite differences.

Finite differences:

- non-symmetric: $f'(i) \approx f(i) - f(i-1)$. (left difference, does not use pixel value $f(i+1)$)
- symmetric: $f'(i) \approx \frac{1}{2}(f(i+1) - f(i-1))$ (O.K. but does not use $f(i)$)
- This can be done by convolution: $f' \approx [-1, +1] * f$, $f \approx [-0.5, 0, +0.5] * f$.

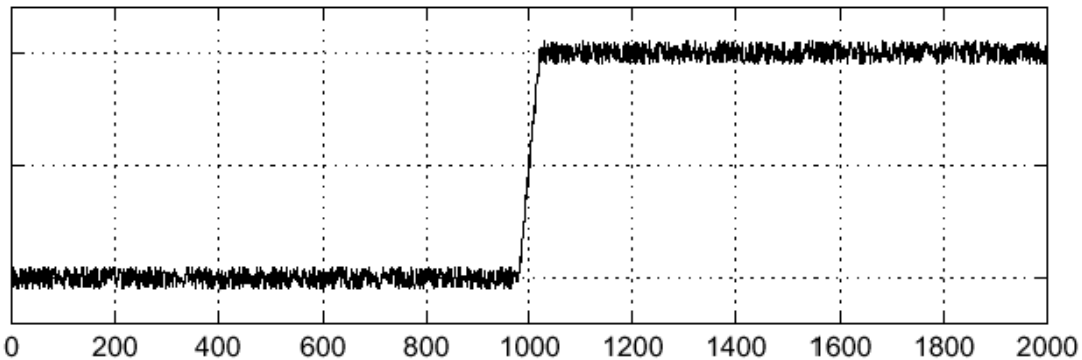
Typical intensity profiles in the pixel neighborhood



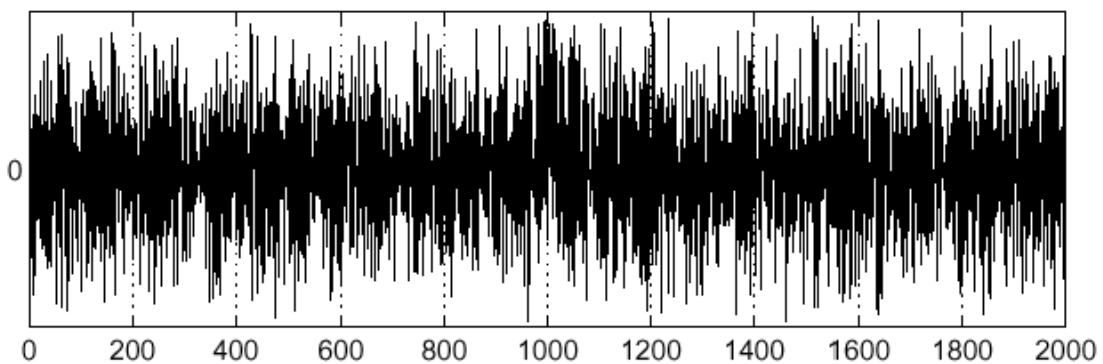
- The first profiles from the left are idealized, i.e. a step transition, a roof transition and a thin line.
- The profile on the right side corresponds to a noise edge. Such edges appear in real images.
- The MATLAB function `improfile` is a useful tool to visualize profiles.

Derivatives are susceptible to noise

image profile with noise

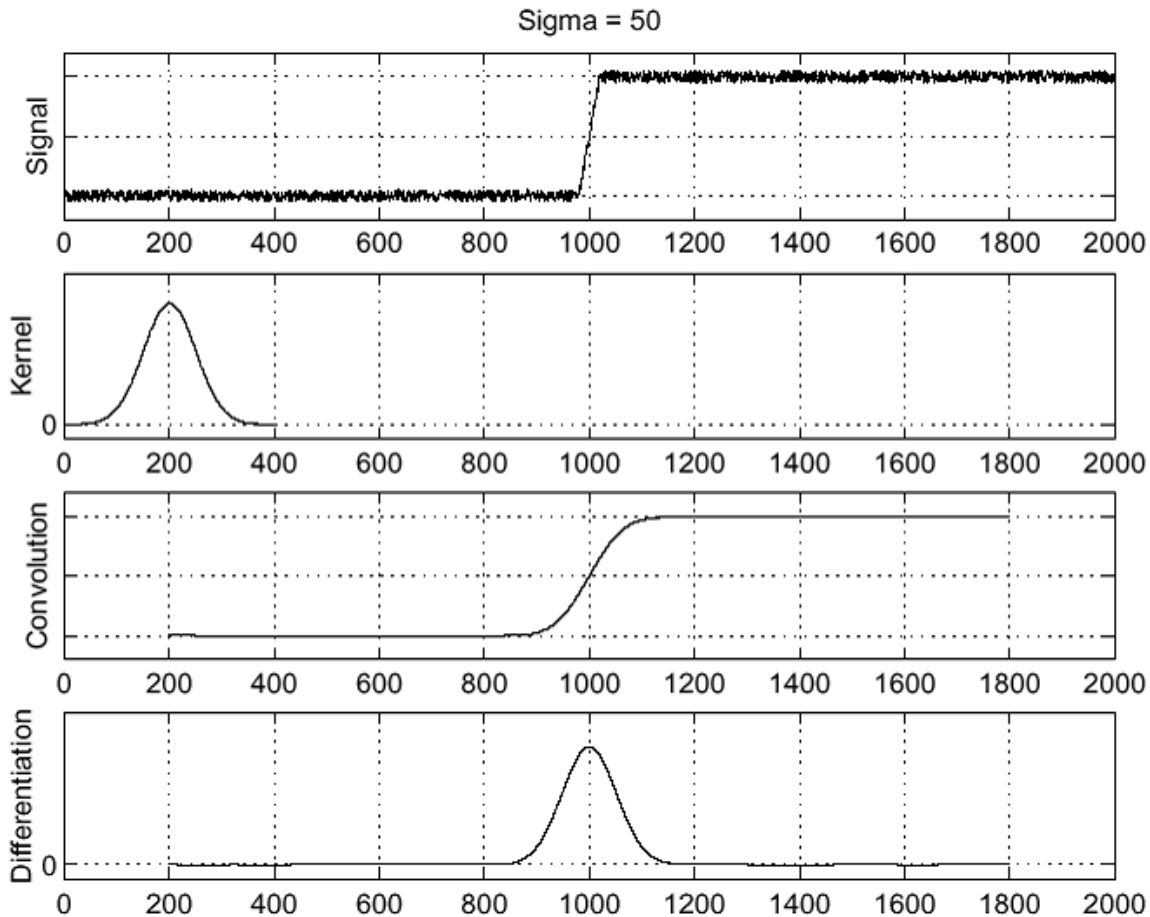


its derivative

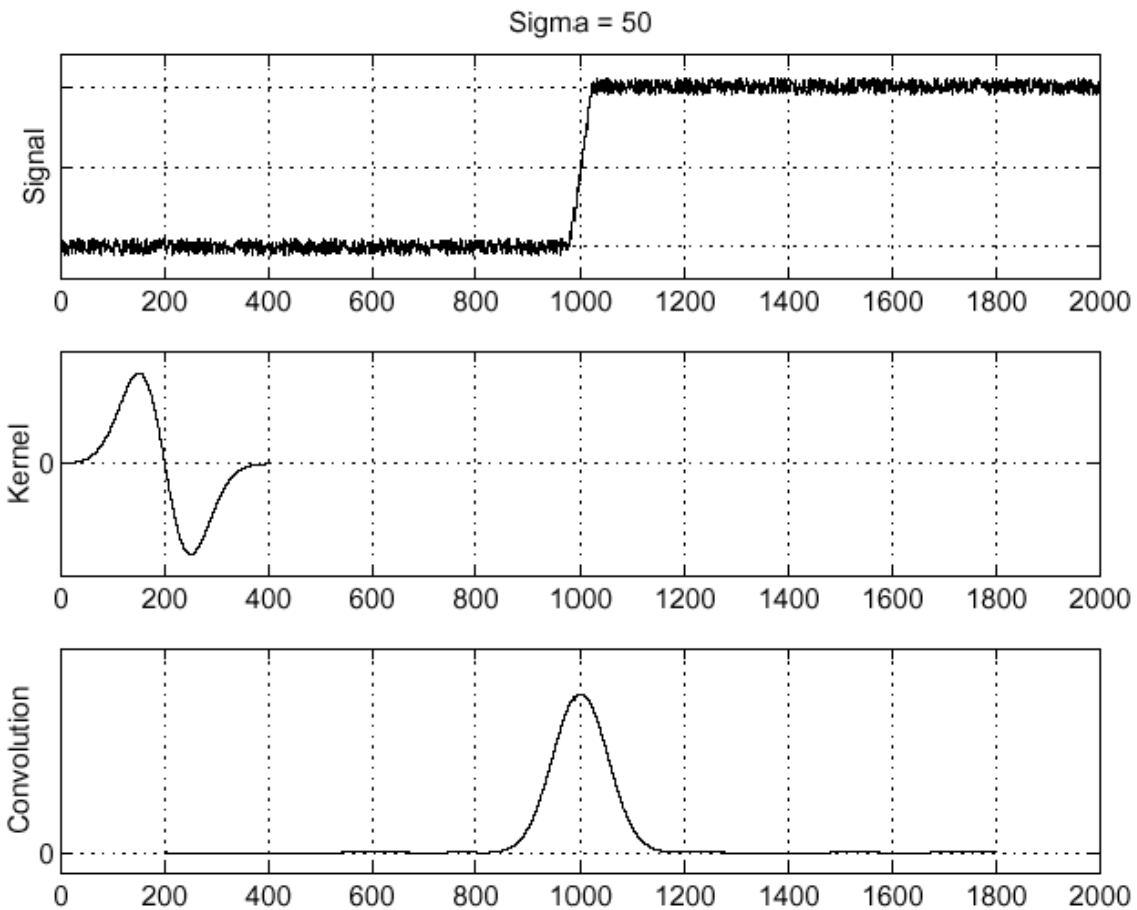


Where is the edgel located in the noisy image?

Image should be smoothed before deriving it



Derivative and convolution interchanged

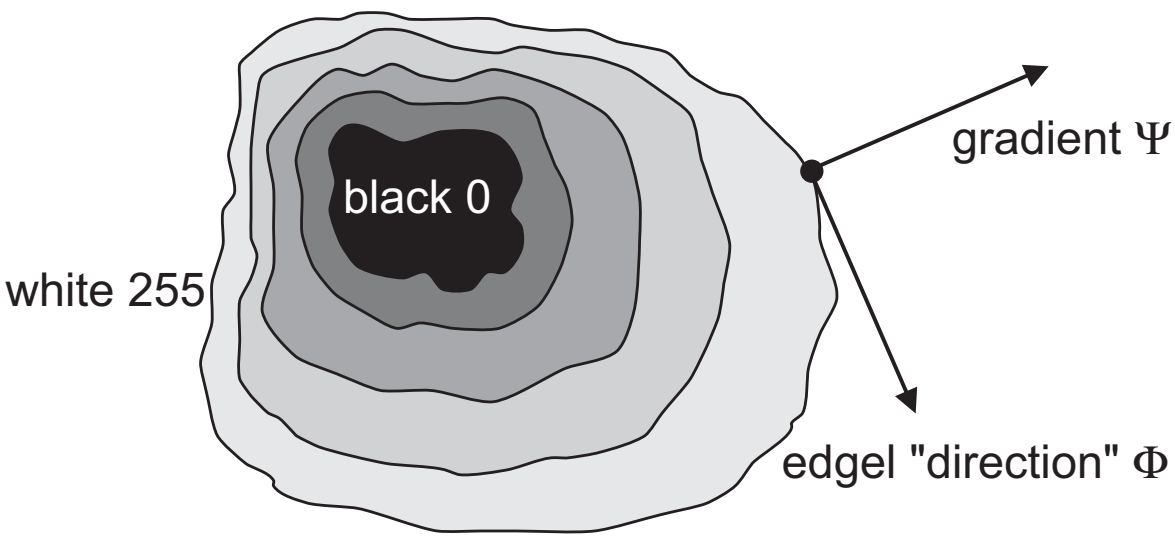


- Two involved operators, the derivative and the convolution, are commutative and can be interchanged.
- Both operations can be combined to a single operator because of their associativity:

$$\frac{\mathrm{d}}{\mathrm{d}x}(h * f) = \frac{\mathrm{d}h}{\mathrm{d}x} * f$$

Edges and object boundaries

- The calculated edges are sometimes used to seek object boundary.
- Provided the object corresponds to a homogeneous region in the image, the border pixels match to edgels.
- Strong edges (edgels) are sometimes chained in order to form object boundaries. This is why the edgel "direction" Φ is defined perpendicular to the gradient direction Ψ.



Convolution 3×3 masks estimating derivative

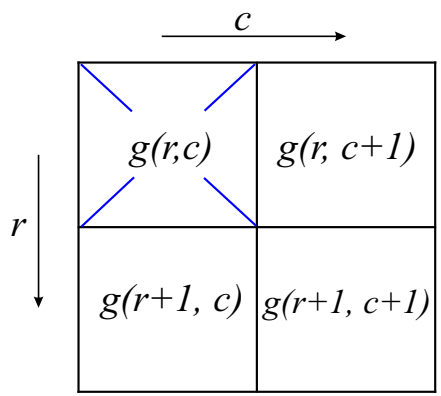
- Roberts (only 2×2), Lawrence Roberts 1963
- Prewitt, Judith Prewitt 1970
- Sobel, Irwin Sobel 1968
- Robinson
- Kirsch, Russell A. Kirsch 1971 and others
- Laplace (approximates the trace of the image function Hessian)

- Eight possible 3×3 masks exist if 8-neighborhood is considered. This quantizes gradient directions in eight discrete values modulo 45^o.
- The image at a particular location is convolved by these eight masks, one by one. The mask response with the highest convolution absolute value determines the edge direction.

Roberts operator in 2×2 neighborhood

Two convolution masks (coordinates: r - row, c - column)

$$h_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \qquad h_2 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}.$$



The magnitude of the gradient is computed as:

$$|\nabla g(r,c)| \approx |g(r,c) - g(r+1,c+1)| + |g(r,c+1) - g(r+1,c)|\,.$$

The disadvantage: a high sensitivity to noise because the neigborhood used for the approximation is small.

Prewitt operator in 3×3 neigborhood

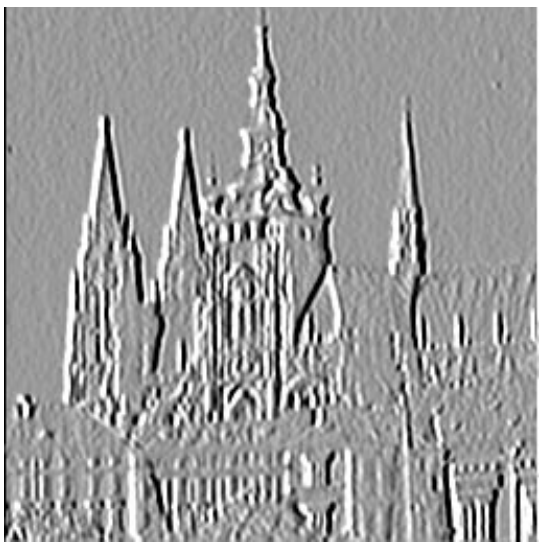
$$h_1 = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}, \; h_2 = \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{bmatrix}, \; h_3 = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix},$$

$$h_4 = \begin{bmatrix} -1 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}, \; h_5 = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}, \; h_6 = \begin{bmatrix} 0 & -1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & 0 \end{bmatrix},$$

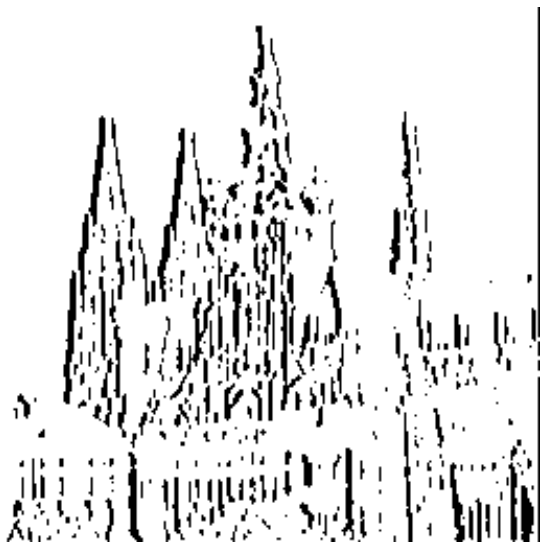
$$h_7 = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}, \; h_8 = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -1 \end{bmatrix},$$

Example: Prewitt operator, western gradients

original 256×256



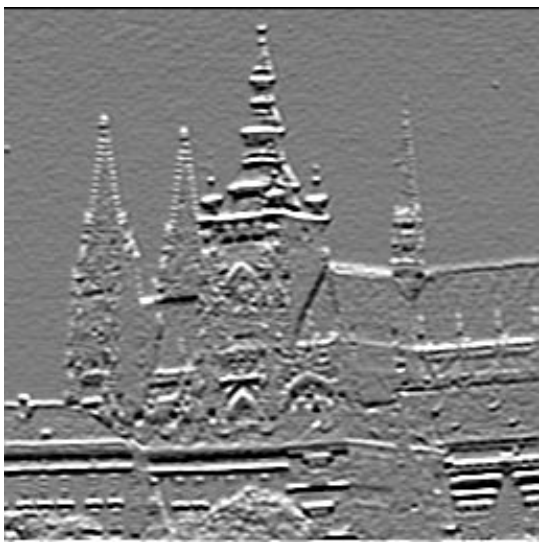
western gradients



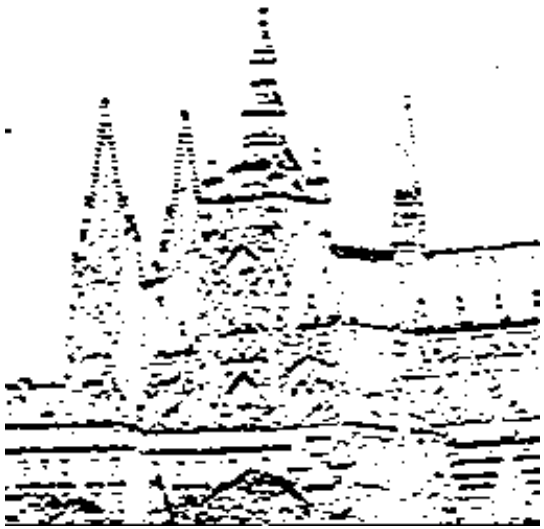
thresholded edges, edgels

Example: Prewitt operator, northern gradients

original 256×256



northern gradients



thresholded edges, edgels

Sobel operator in 3×3 neigborhood

$$h_1 = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}, \qquad h_2 = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix},$$

$$h_3 = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \ldots$$

Robinson operator in 3×3 neigborhood

$$h_1 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & -1 & -1 \end{bmatrix}, \qquad h_2 = \begin{bmatrix} 1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & -1 & 1 \end{bmatrix},$$

$$h_3 = \begin{bmatrix} -1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & 1 & 1 \end{bmatrix}, \ldots$$

Kirsch operator in 3×3 neigborhood

$$h_1 = \begin{bmatrix} 3 & 3 & 3 \\ 3 & 0 & 3 \\ -5 & -5 & -5 \end{bmatrix}, \qquad h_2 = \begin{bmatrix} 3 & 3 & 3 \\ -5 & 0 & 3 \\ -5 & -5 & 3 \end{bmatrix},$$

$$h_3 = \begin{bmatrix} -5 & 3 & 3 \\ -5 & 0 & 3 \\ -5 & 3 & 3 \end{bmatrix}, \ldots$$

Laplacian of the image function

Image function $f(x, y)$, its Laplacian is:

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

- $\nabla^2 f$ is a scalar, not a vector. There is no "direction" provided.
- The Laplace operator is rotation-invariant.
- For a monotonically increasing image function $f(x, y)$, Laplacian crosses zero at a place where the gradient $\|\nabla f(x, y)\|$ attains its maximum.

Discrete approximation of Laplacian

- The second finite difference is computed from first finite differences:

$$\frac{\mathrm{d}^2}{\mathrm{d}x^2} \approx [-1, +1] * [-1, +1] = [+1, -2, +1]$$

- Laplacian is the sum of finite differences in horizontal and vertical directions:

$$\nabla^2 \approx \begin{bmatrix} 0 & 0 & 0 \\ 1 & -2 & 1 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \\ 0 & -2 & 0 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

- Alternatives:

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \quad \begin{bmatrix} 2 & -1 & 2 \\ -1 & -4 & -1 \\ 2 & -1 & 2 \end{bmatrix}, \quad \begin{bmatrix} -1 & 2 & -1 \\ 2 & -4 & 2 \\ -1 & 2 & -1 \end{bmatrix}$$

Sharpening by a Laplacian

Sometimes we do not aim at detecting edgels. Instead, edges should be enhanced (sharpened). Laplacian as an omnidirectional high-pass filter is suitable.

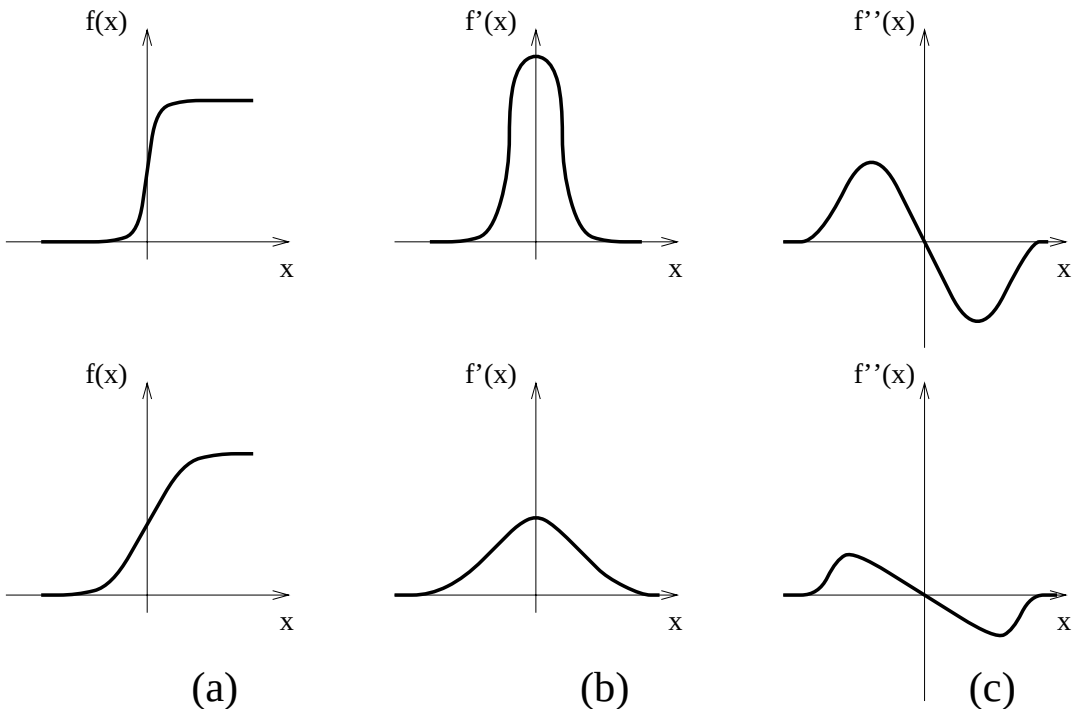
Original 256×256

Edges (Laplace)

Sharpening (- 0,4 $\cdot$)

Edgel detector based on 2nd derivative (Marr-Hildreth)

- The extremum of the first derivative of a function of a single variable (a 1D signal) matches the location in which the second derivative passes the zero level.



- For functions of two variables, e.g. a 2D image it is not the same: Laplace operator ∇^2 comes into the play.

Derivation, Laplacian of Gaussian operator (LoG)

Laplacian $\nabla^2 f(x,y)$, recall the slide 25, is even more sensitive to noise than the gradient $\Rightarrow$ It is combined with a Gaussian G again. The two operators can be combined to one $\rightarrow$ LoG (Laplacian of Gaussian).

$$\nabla^2(G * f) = (\nabla^2 G) * f = \mathrm{LoG}(f)$$

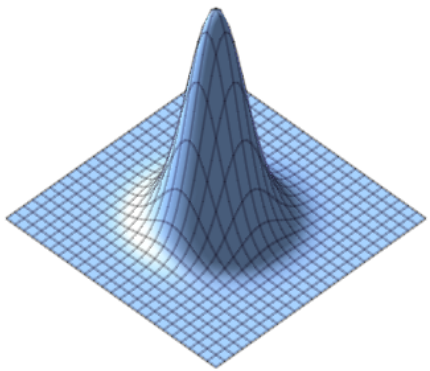
For a given σ, (substitution due to rotational symmetry $x^2 + y^2 = r^2$):

$$G(r) = e^{-\frac{r^2}{2\sigma^2}}, \quad G'(r) = -\frac{1}{\sigma^2}\, r\, e^{-\frac{r^2}{2\sigma^2}}, \quad G''(r) = \frac{1}{\sigma^2}\left(\frac{r^2}{\sigma^2} - 1\right) e^{-\frac{r^2}{2\sigma^2}}.$$

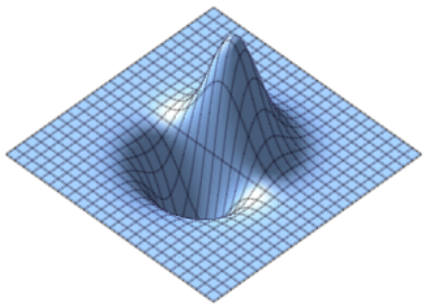
(c is normalization constant)

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

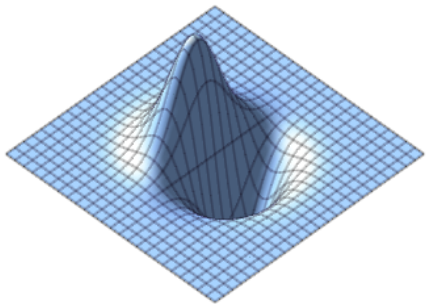
2D operators we came across



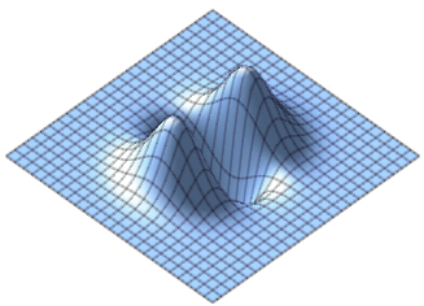
Gaussian



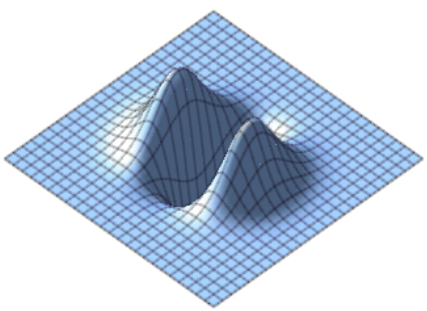
$\frac{\partial}{\partial x} G$



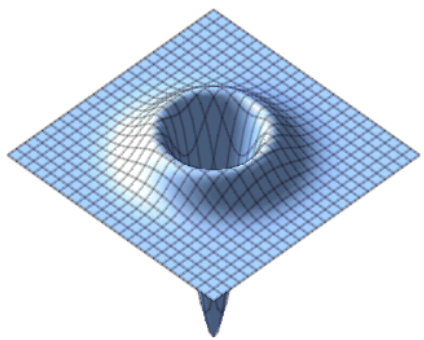
$\frac{\partial}{\partial y} G$



$\frac{\partial^2}{\partial x^2} G$



$\frac{\partial^2}{\partial y^2} G$



Laplacian of Gaussian

Difference of Gaussians (DoG) as the approximation of LoG

- The aim is to approximate LoG operator, i.e. $\nabla^2 G$.
- The difference of two images created by a Gaussians with two different σ provides a rough approximation o LoG.

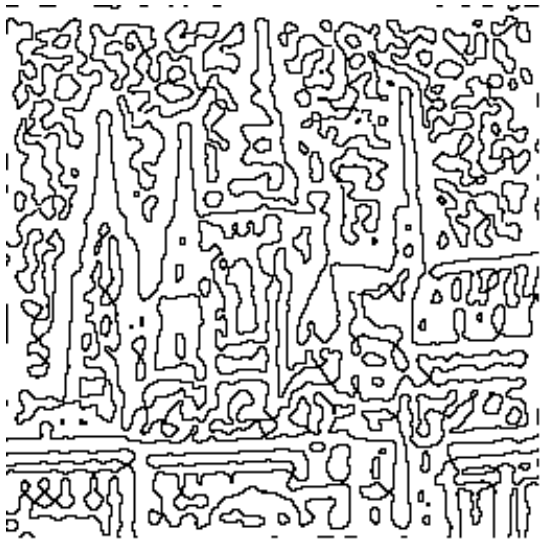
How to evaluate zero crossings?

- While implementing zero crossing, it is recommended to avoid a naïve solution, i.e., thresholding of LoG in the interval close to zero. The discontinuous edgels would be obtained.
- It is better to really detect sign changes, e.g. in a 2×2 mask with representative point in the top-left corner. The edgel is detected if there is a sign change inside the image.

Zero crossings: example

Original

DoG $\sigma_1 = 0,1$ $\sigma_2 = 0,09$

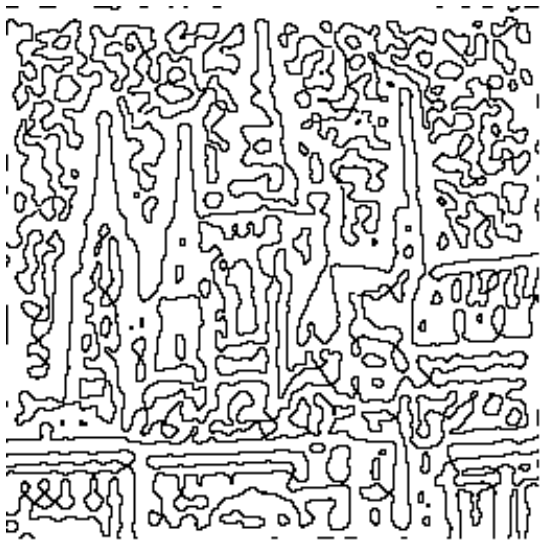


Zero crossings

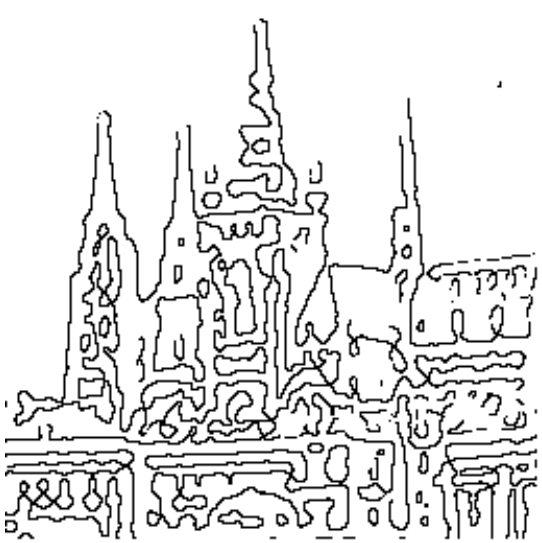
Disadvantages:

- Sharp shapes are blurred to much. E.g. sharp corners are lost.
- Edgels are connected into closed curves $\Longrightarrow$ "A plate of spaghetti".

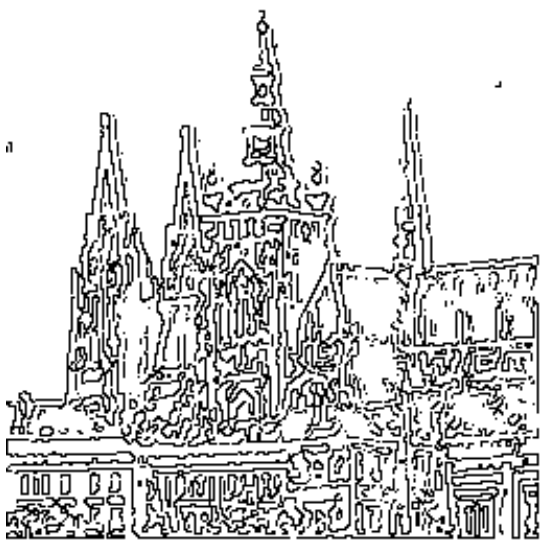
Thresholding according to the strength



zero crossings



after weak edgels removal



LoG, $\sigma = 0.2$

LoG & physiology

- The retina evolved from the brain. The retina senses light (rod and cones) and also performs preprocessing. There is about 1:100 data reduction $\Leftarrow$ reduced transmission capacity of the optical nerve.
- Circular receptive fields. Their outer part contributes to the response by a negative sign than the inner part (so called center-surround arrangement).

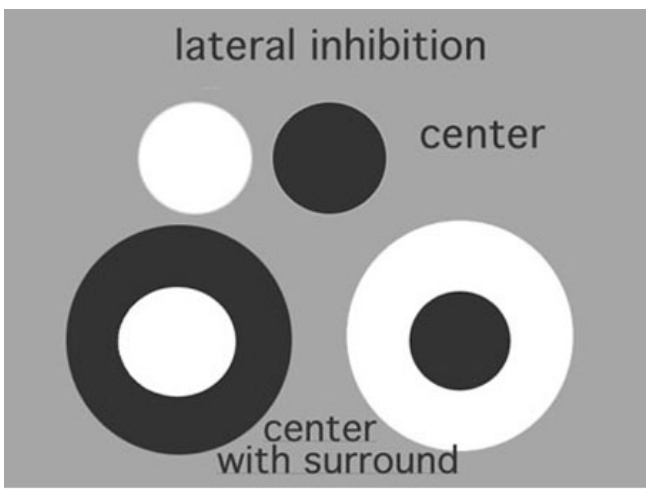


Fig. 10. Center-surround receptive fields can be ON center or OFF center with the oposite sign annular surround.

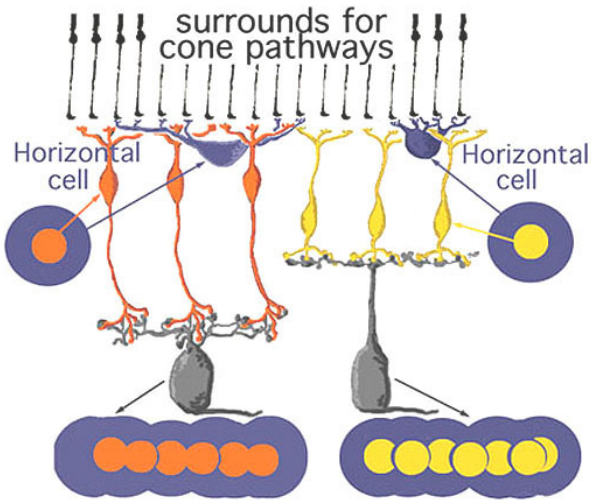
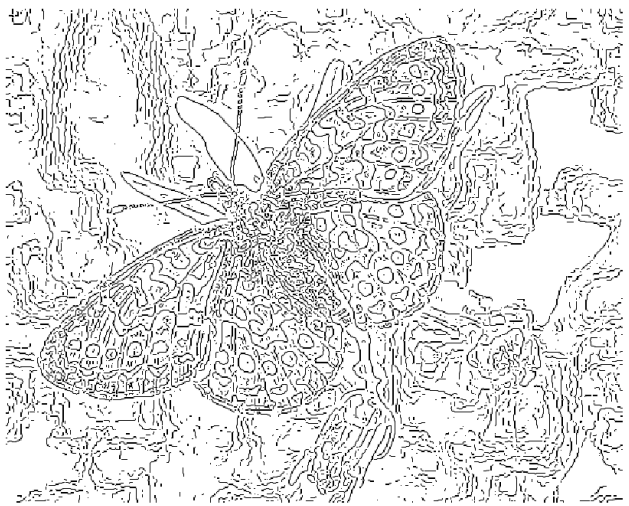


Fig. 12. Diagram of the organization of center-surround responses using horizontal cell circuitry to provide the antagonistic surround.

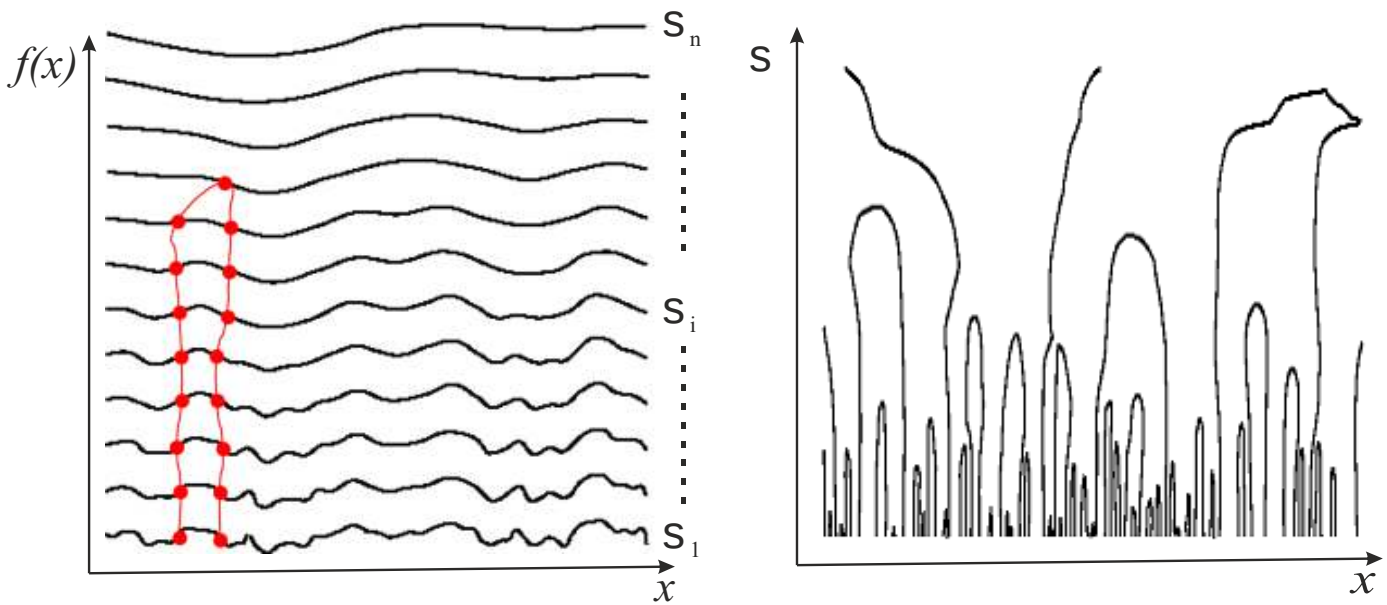
The smoothing scale selection issue (1)



The smoothing scale selection issue (2)

- How to chose Gaussian σ while calculating the derivative? The bigger σ, the ...
  - better noise suppression;
  - more weak edges vanish;
  - smaller eges localization precision.
- This issue is not constraint to edge detection only. It is a general issue appearing while detecting features based on local properties of the image function, e.g. salient points as Harris corners.
- We are often not interested in details even they did not appear due to noise. Imagine we wish to look at the image from the 'bigger distance' and detect only more important edgels or other primitives.

Scale space (for 1D signal)



- While the scale grows (σ increases) the edgel can vanish by being connected to other edgel. However, a new edgel is not created.
- In 1D: A.P. Witkin: Scale-space filtering. Proceedings of 8th Int. conference on AI, August 1983, pp. 1019-1022.
- In 2D: T. Lindeberg: Scale-Space Theory in Computer Vision, Kluwer Academic Publishers/Springer, Dordrecht, Netherlands, 1994.

Canny edgels detector

- A simple edgels detector crowning the efforts for the best edge detector.
  J. Canny: A Computational Approach To Edge Detection, IEEE Transactions on Pattern Analysis and Machine Intelligence, 8(6):679–698, 1986.
- Formulated as the seek for the optimal filter under practically needed constraints.
- It is used in many applications.

- Input: a gray-scale image.
- Output: a binary image with edgels.

Canny edgels detection algorithm:

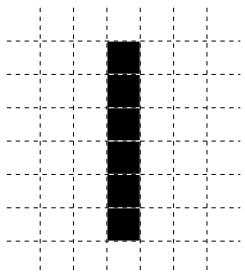
1. Compute gradient directions.
2. For each pixel, compute smoothed 1D directional derivative in the gradient direction.
3. Find the magnitude maxima of these derivatives.
4. Get edgels by thresholding with hysteresis.
5. Performs the synthesis of edgels obtained for different smoothing (simple implementations do not perform this synthesis step).

Optimal linear filter for edgels detection

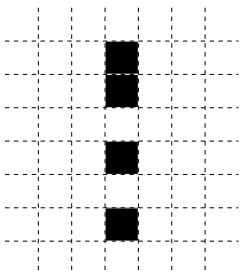
A simplified model is assumed: the ideal step edge; additive Gaussian noise independent on the image.

What do we want to maximize (=optimization constraints)?

1. Reliable detection (find almost all existing edgels);
2. Good localization (small error in the edgel position);
3. Unique responses (minimum nonexisting edgels found).

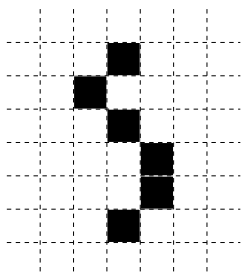


real edgel

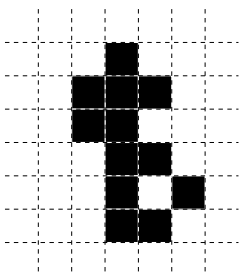


unreliably detected

- Contradicting requirements: The better detection the worse localization.
- The 'best' compromise is sought: the product of criteria 1 and 2 is optimized and, finally, the criterion 3 is added (details are omitted here for brevity).
- The result cannot be written using a single formula. However, its spirit is very similar to the derivative of the Gaussian.



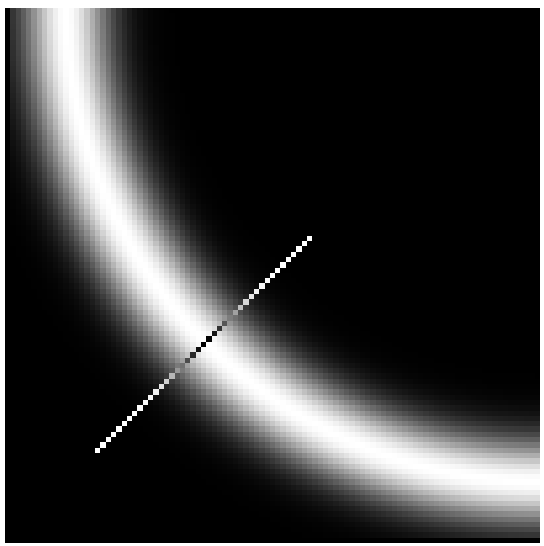
badly localized

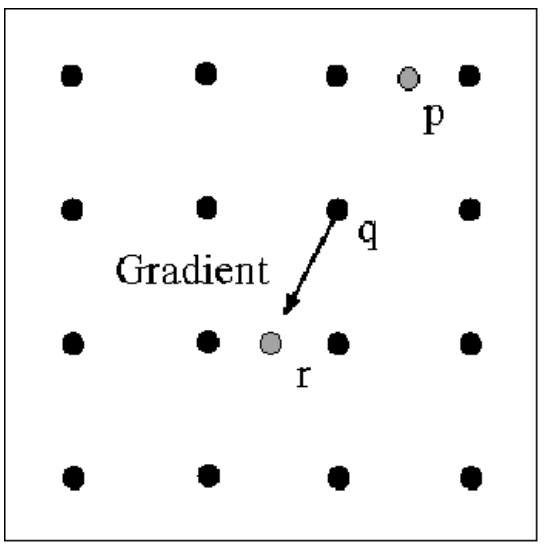


ambiguously detected

Finding the gradient maxima in 2D

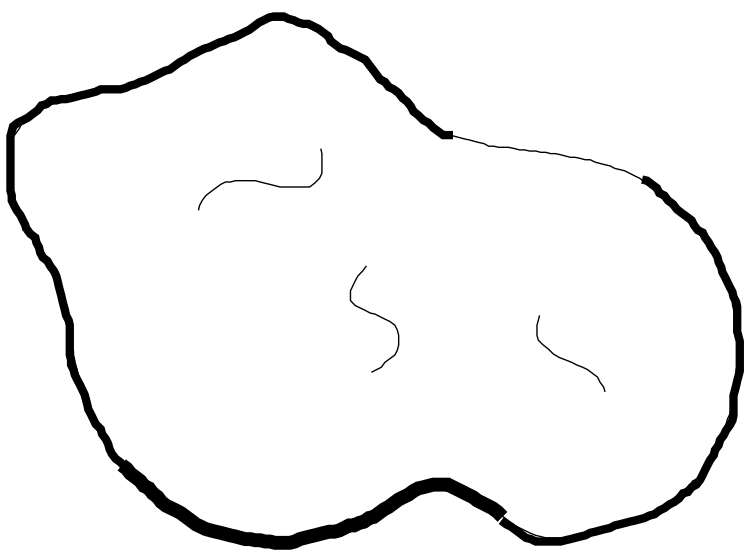
- 1D maxima are sought in an approximate direction of the gradient.
- The image function in the gradient direction is sampled by a grid.





Canny edgels by thresholding with hysteresis

- Why? We want to suppress short (usually unimportant) chains of edgels. At the same time, we aim at avoiding fragmentation of long edgel chains likely corresponding to object boundaries.
- This cannot be done by thresholding. The trick is to use two thresholds $T_1 < T_2$. Edges stronger than T_2 are automatically edgels. Edgels stronger than T_1 are linked if connected to stronger edges.



Canny edgels, example; threshold = 0.15, three σ values

original

thres = 0.15, σ = 0.2

thres = 0.15, σ = 0.5

thres = 0.15, σ = 1.0

Canny edgels, example; threshold = 0.3, three σ values

original

thres = 0.3, σ = 0.2

thres = 0.3, σ = 0.5

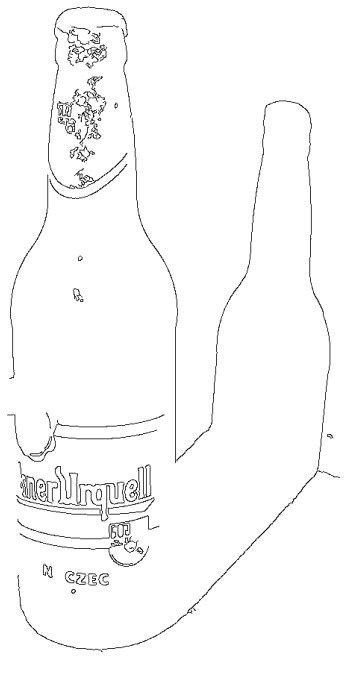
thres = 0.3, σ = 1.0

Canny edgels, example; threshold = 0.5, three σ values

original

thres = 0.5, σ = 0.2

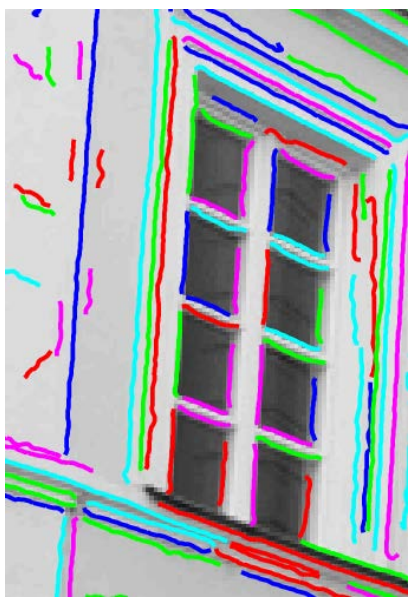
thres = 0.5, σ = 0.5



thres = 0.5, σ = 1.0

Chaining edgels to lines to be used for windows detection

- Filtered by Laplace
- Detected zero-crossings
- Chained to lines



Courtesy: Radim Šára

Criticism of edges at corners

- Edge detectors (e.g. LoG, Laplacian of Gaussians) are not precise ‘at corners’.
- Edge detectors do not serve for finding corners, though.
- We have corner detectors for detecting corners (see a separate lecture).