4. EDGE DETECTION

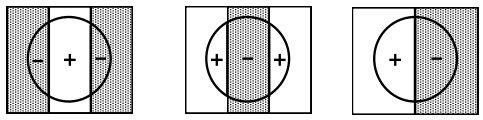
Edges - important image features

Object boundaries, or edges, show up as intensity discontinuities in an image. Edges are important in human vision: our visual system can recognise objects from their crude outlines. Detection of object boundaries in images, in a way that is consistent with human vision, was an important research topic for a long period. Currently there exist many well established algorithms for edge detection and, importantly, their limitations are well understood.

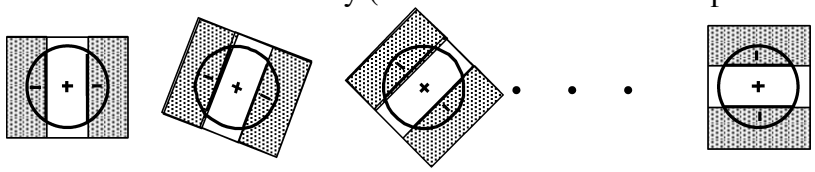
Edge detection in the visual cortex

Visual cortex (part of the brain processing visual inputs) is divided into a number of layers, starting from layer 1 close to the surface. Certain cells in layer 1 of the visual cortex (the so called *simple cells*) are known to respond to brightness discontinuities, i.e. edges. There are several types of specialised cells, each type responding specifically to:

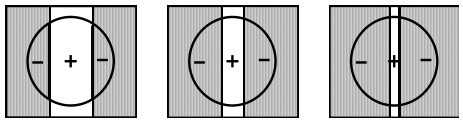
- a particular *type* of the discontinuity



- *orientation* of the discontinuity (illustrated below on example of the bar feature)



- *spatial frequency* of the discontinuity



Detection is accomplished by polling together excitatory and inhibitory responses from the appropriately aligned cells forming an earlier part of the visual pathway (see the exercises).

Edge detectors employed in computer image analysis are modelled on the responses of these cortical simple cells.

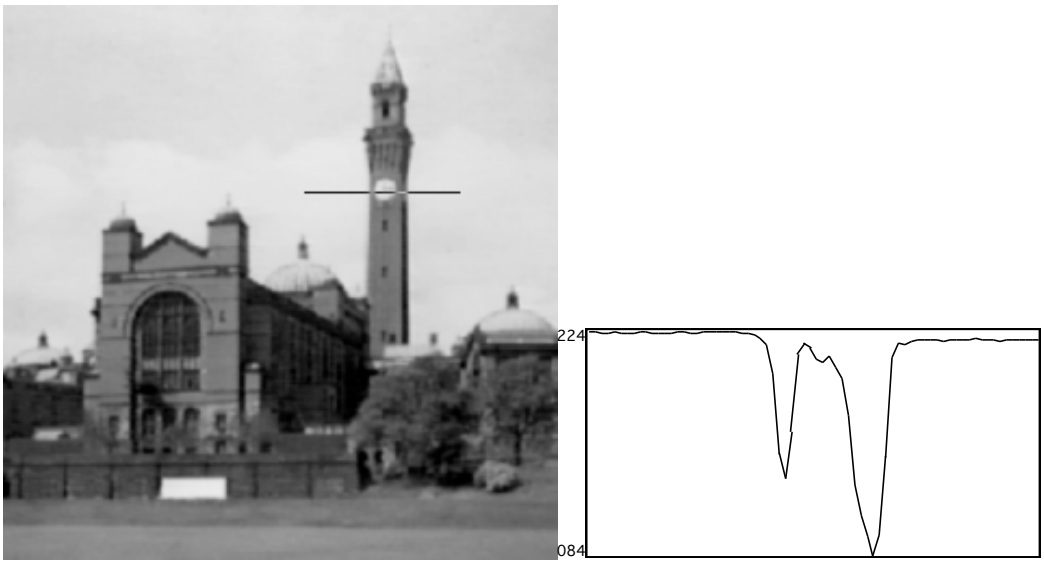
Edge detection in digital images

Edge detection (also called edge extraction) operations normally produces a *parametric (intrinsic)* image which contains *only* edge information, i.e. intensity information is lost. That information is then used by higher-level algorithms for feature extraction and/or object recognition, or for visual edge enhancement by combining an edge image with the original image.

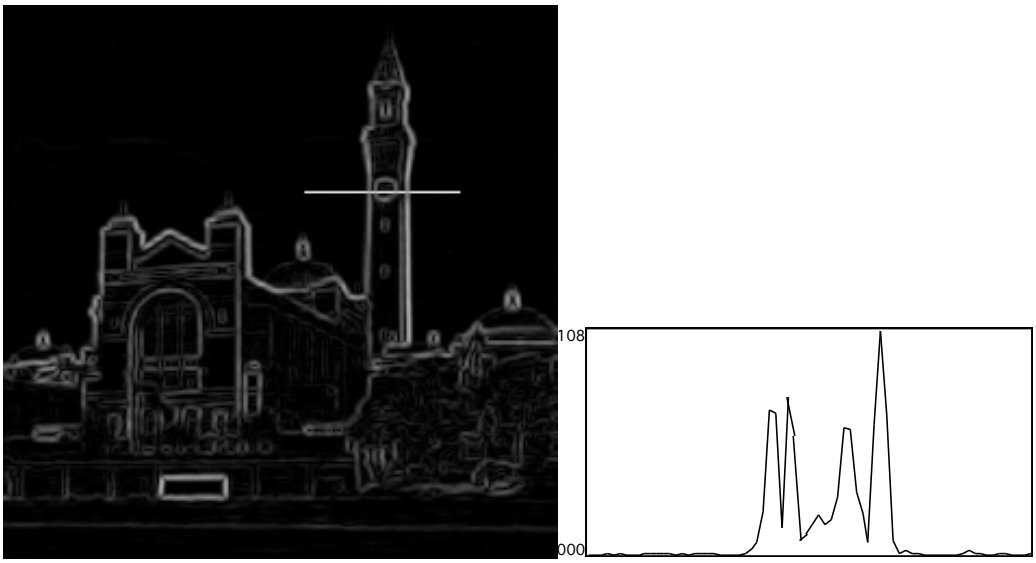
A parametric image shows *local* grey level discontinuities, which subsequently can be composed into more elaborate boundary by higher level processing routines, using higher level information (e.g. a model). A local edge is a small area in the image where the local grey levels are changing rapidly. It is characterised by *high -frequency components*: local

grey levels are changing rapidly in a small area of the image. Spot these areas and changes in both the image and a line profile below.

Grey level image and its line profile

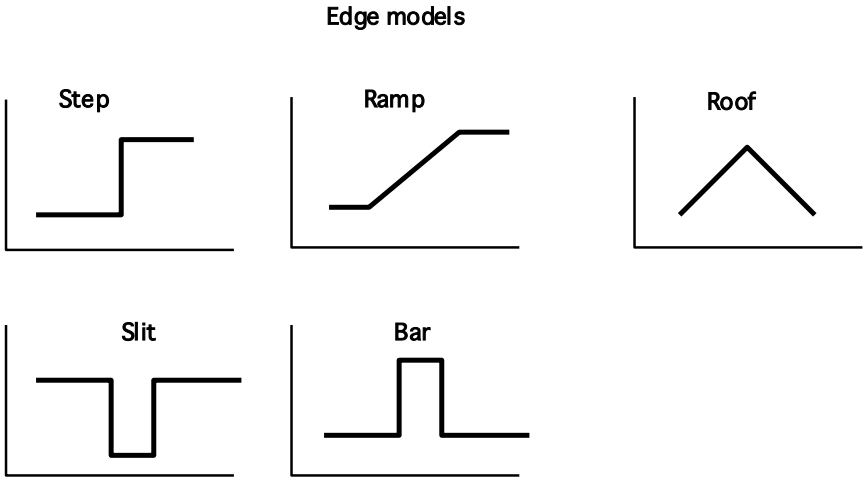


Edge image (derived from the grey level image above) and its line profile



Compare the values of the corresponding pixels in the original image and the edge image. How are they related?

Edges in images are normally classified into the following different types (models), illustrated below by the line profiles:



Edge detection by spatial filtering

In analogy to the cortical cells, edges in digital images are detected by polling "excitatory" (positive) and "inhibitory" (negative) "responses" within small local neighbourhoods. Different edge models will employ different ways of combining these positive and negative "responses" together. However, very elegantly, all these different edge types can be detected using the same computational method: convolution. The different edge types will use different convolution kernels (also referred to as *edge operators*). Most edge operators compute (directly or indirectly):

- edge direction - aligned with the direction of maximum grey level change
- edge magnitude - measure of that change (normally related to grey level values)

The edge operators are normally classified according to the underlying mathematical (differential) model. The most common types include

- difference
- gradient
- Laplacian

Difference operator

A difference operator is an example of *directional operator*; it allows to extract vertical, horizontal and diagonal edges independently. For example, to detect the presence of vertical edges:

- shift an image to the left by one pixel
- subtract from original

This produces a brightness difference

$$O(x,y) = I(x+1,y) - I(x,y)$$

If pixels at (x,y) and (x+1,y) have similar brightness, a low output value will result; dissimilar brightness (edge) will result in a high output value. Edges in other orientations are detected analogously (see the exercises).

The difference operator can have a number of implementations, of which the second is preferable:

- directly by shifting and subtracting

- by using filters

Both implementations correspond to convolution of a discrete image with the kernels of the form:

0	0	0
-1	1	0
0	0	0

Vertical

0	-1	0
0	1	0
0	0	0

Horizontal

-1	0	0
0	1	0
0	0	0

Vertical & horizontal

Properties of the difference filters:
- sensitive to noise
- highlight only dark-to-light edges in a single direction
- light-to-dark edges yield negative values and are normally set to 0 (black)

In addition to calculating the *magnitude of the edge* (as above), *direction* image can be created by calculating the direction value as:

$$\varphi = \max_{(x,y)} \{ \tan^{-1}(\frac{\Delta I}{\Delta x} / \frac{\Delta I}{\Delta y}) \}$$

(NB symbol delta Δ, denotes the difference, e.g. $\Delta x = x_i - x_j$)

Gradient operator

A gradient operator is also a *directional (anisotropic) filter*. Up to 8 gradient images can be generated:

N, NE, E, SE, S, SW, W, NW

Each detect edges aligned in one particular direction.

Mathematically the gradient image function is defined in terms of directionally oriented spatial derivatives:

$$df / dx = [f(x + \Delta x, y) - f(x,y)] / \Delta x$$
$$df / dy = [f(x, y + \Delta x, y) - f(x,y)] / \Delta y$$

Gradient operator is a filter because:
- it attenuates completely low frequency components
- it passes through high frequency components
- constant brightness sequences turned into 0

Output value is a measure of *edge strength (gradient magnitude)*, i.e. difference between neighbouring intensities in a particular direction. The edge location is taken to be at the maximum of the response.

The gradient operator detect edges in a particular direction. Very often these directional responses are combined together, for example by taking the maximum over the directional responses. The resulting *omni-directional operators* are referred to as *non*-linear filters (think why?). Examples of such operators are shown below.

Sobel operator

-1/4	0	1/4
-1/2	0	1/2
-1/4	0	1/4

-1/4	-1/2	-1/4
0	0	0
1/4	1/2	1/4

Max

Roberts operator

0	0	0
-1	0	1
0	0	0

0	1	0
0	0	0
0	-1	0

divide by 2

0	0	1
0	0	0
-1	0	0

1	0	0
0	0	0
0	0	-1

divide by ¦2

Properties of the gradient filters:
- negative differences turned into 0
- can selectively analyse edges in a particular direction
- but: does not show all the edges at once
- only 8 directions available
- two outputs can be generated: edge gradient and edge direction

Laplacian operator

Laplacian operator is an example of *omni-directional* (*isotropic filter)*. Mathematically it can be expressed by the equation:

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

The discrete approximation is the sum of the second spatial derivatives which can be represented by the 3x3 window operator

Laplacian

0	1	0
1	- 4	1
0	1	0

Properties of the Laplacian filter:
- unidirectional operator (i.e. enhances edges regardless of their orientation)
- attenuates completely low frequency components
- passes through high frequency components
- a large coefficient surrounded by smaller positive and negative values
- constant brightness and linearly changing brightness sequences turned into 0

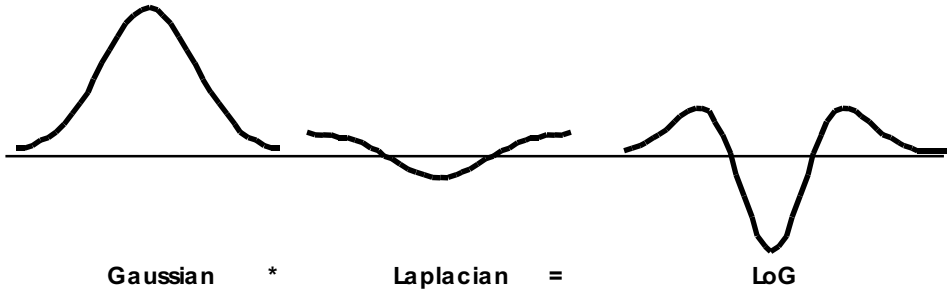
- highlights both positive and negative differences
- highlighting is sharper than for gradient operators
- does not provide information about orientation
- doubly enhances noise in the image

Smoothing and Laplacian

All the described edge detection filters are sensitive to noise. Laplacian-of-Gaussian (LoG) operator smoothes the image using a Gaussian filter, to damp high-frequency components, and then Laplacian is applied.

The size of Gaussian determines the spatial scale of the edges:
- small Gaussian (little smoothing) will enable the detection of fine edges
- large Gaussian (severe smoothing) will help to detect coarse edges

Gaussian filter can be combined with Laplacian filter into a single "**L**aplacian **o**f **G**aussian" (LoG) operator.

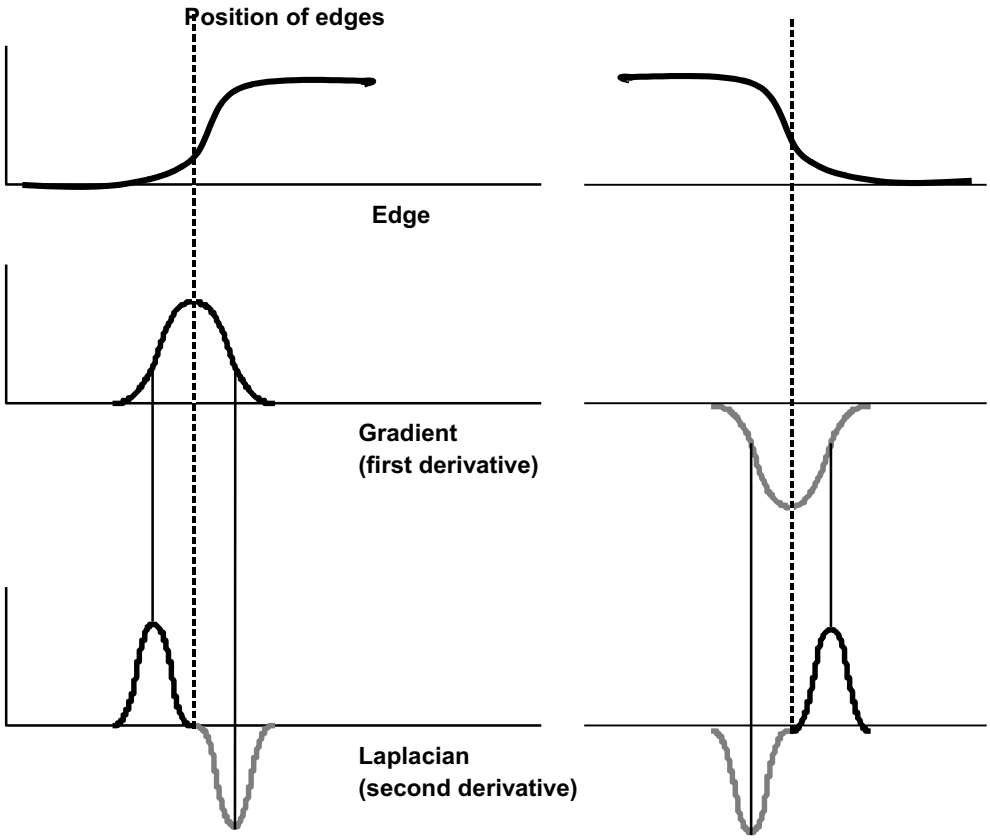


A computationally similar filter can be constructed by subtracting two images obtained by filtering the original with two Gaussians with different smoothing constant, this filter is called DOG (**D**ifference **O**f **G**aussians), see the exercise.

Zero crossings of second spatial derivatives

The second derivative of a function changes sign (i.e. crosses the axis, or zero) at the spatial location corresponding to the change of curvature, or the "midpoint" of the edge. In images this occurs in all directions except in the direction parallel to the edge. This property is exploited to determine edge orientation. The figures below illustrate the relationship between the edge location and the zero crossing of its second spatial derivative.

Please also note the results of the first derivative, where black lines indicate positive responses and grey lines – negative. Depending on the needs, negative responses can either be set to zero or turned to positive (e.g. by taking the absolute value of the response). Think: why you should want to take one or the other approach? Can you see why the zero crossing is better than the maximum of the first derivative?



Notes:

Post-processing

An edge detection operator produces an image in which a pixel value (an *edge response*) is proportional to the contrast at the edge location (some operators produce negative values – think why?). In many cases we are only interested in strong edge responses, associated with major edges, and want to ignore weak responses, normally associated with noise and inessential detail. A common post-processing operation after edge detection is thresholding, where weak edge responses are removed and only strong ones are retained. Selecting a suitable threshold value can be difficult (what is a typical shape of histogram in an edge-detected image?).

Canny edge detector

Canny proposed an edge detecting operator which, in addition to simply detecting all the edges, carries out an optimisation process to ensure that only "important" edges are shown, that they are precisely localised, and that there is a single detector response for a single edge.

The steps of Canny edge detector algorithm:

- Convolve image I(x,y) with a Gaussian function $G(x,y,\sigma)$, to obtain smoothed image
  $$S(x,y) = I(x,y) * G(x,y,\sigma)$$
  The degree of smoothing depends on parameter σ

- For each pixel in the image find the direction of the largest gradient:
  $$G(x,y,\varphi) = \partial S(x, y) / \partial \varphi$$
  $$\varphi(x,y) = \max(\tan^{-1} (G(x,y)))$$
  and, for this direction, store the value of the gradient:
  $$G(x,y) = \max(G(x,y,\varphi))$$

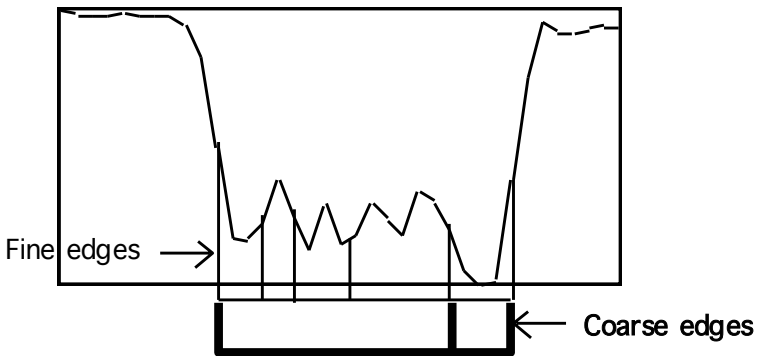
- Find accurate location of edges using the *maximum suppression*: -
  - for each pixel examine gradient values in direction of the gradient f within a certain small neighbourhood; set to 0 all non-maximum values in that neighbourhood.

- Threshold the remaining edges so that only the strongest edges are retained.

Multi-level (hierarchical) analysis

Human visual system works on different levels of resolution. Normally processing proceeds from coarse level to fine level. Such approach is computationally efficient. Processing involves the creation of a sequence of images at various spatial resolutions, spatially related to each other. A *scale-space* representation of an image may be formed by convolving the image with a sequence of filters containing a scale parameter.

A consistent scale-space filter has a property that going from low to high resolution images existing features normally do not disappear, but new ones may appear.



Spatially related sequence of images at increasing resolution can be thought of as a pyramid, with the original (high resolution) image at the bottom of the pyramid. Applications include segmentation, feature extraction, description and matching.

Scale-space filtering for edges

Edges occur on different scales. The size of Gaussian used for smoothing determines the spatial scale of the edges

- small Gaussian will preserve higher frequency components and enable the detection of fine edges
- large Gaussian will remove higher frequency components enable detection of coarse edges

A set of Laplacian of Gaussian filters, with different scale parameter σ, is applied and their results re-combined to extract edges at different spatial scales.

Further reading and exploration

Sonka, M. Hlavac, V. Boyle, R. (1993 / 1999) *Image Processing, Analysis and Machine Vision*, Chapman & Hall Computing, sec. 4.3.
Gonzalez, R.C. & Woods, R.E. (1992) Digital Image Processing, Addison-Wesley, sec. 4.3.

HIPR

Worksheets->Feature detectors:
- Roberts Cross Edge Detector
- Sobel Edge Detector
- Canny Edge Detector
- Compass Edge Detector
- Zero Crossing Detector
- Line Detector

Digital Filters:
- Frequency Filters
- Laplacian/Laplacian of Gaussian Filter
- Unsharp Filter - edge enhancement filter

CVIP

Analysis -> Edge/Line detection (first three groups of operators)
Utilities -> Filter

Exercises

1. Using the symbols introduced in Unit 3(2), representing receptors, cells, and excitatory and inhibitory responses, draw diagrams corresponding to two-dimensional bar detectors, slit detectors and edge detectors.
2. Display image **bham.tif** and look for instances of various edge models.
3. Specify convolution kernel coefficients of a gradient filter for detecting edges in NW direction.
4. Apply several types of edge operators to an image and compare the operator responses.
5. Display a histogram of an edge-detected image. Can you see a sensible place at which to threshold the image to retain only strong edges?
6. Using a sequence of CVIP operations, compute DOG of an image. Compare it with LOG.