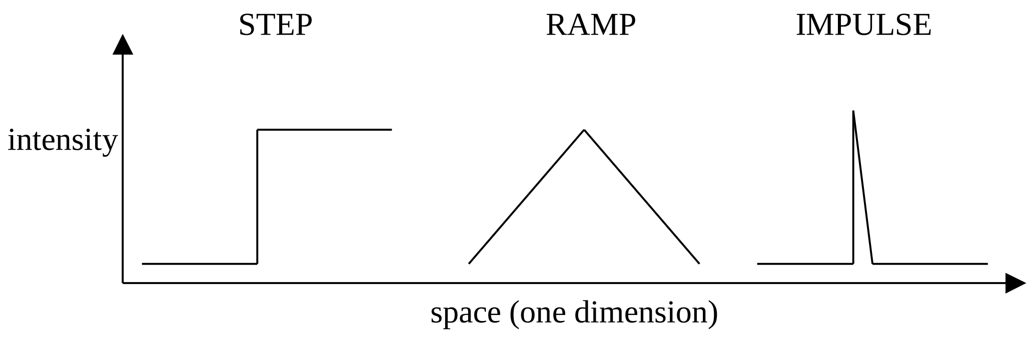
Lecture notes: Edge Detection

In the reconstruction approach (Marr's theory), edges play an important role. They are probably the most prominent features visible in an image and therefore contribute heavily to the so-called primal sketch (early understanding of the image).

What is an edge? It is an intensity gradient. Several common examples are shown in one-dimension (for simplicity) below:



In two dimensions, the gradient will have a magnitude and a direction:

0	0	10	10
0	0	10	10
0	0	10	10
0	0	10	10

Here we have an image area with a vertical edge. We can compute the gradient magnitude and direction as:

$$\left|grad\,(I[r,c])\right| = \sqrt{\left(\frac{\partial I}{\partial r}\right)^2 + \left(\frac{\partial I}{\partial c}\right)^2}$$

and

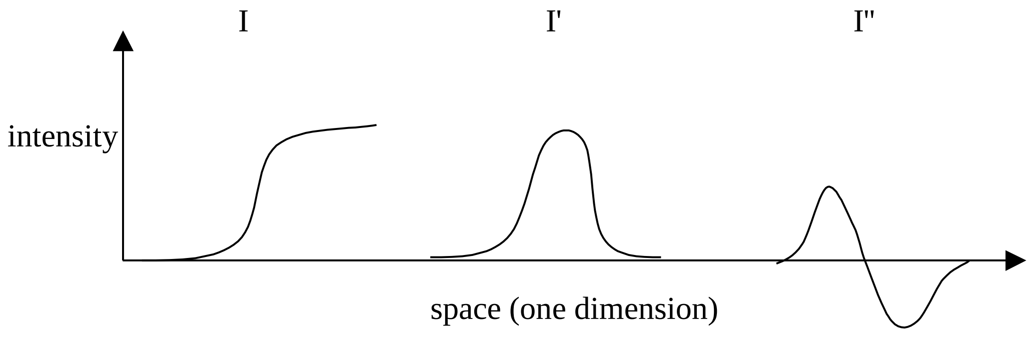
$$\theta = \tan^{-1}\left(\frac{\partial I / \partial r}{\partial I / \partial c}\right)$$

Note that the magnitude measured is the absolute value of the actual magnitude (e.g. could be step up or step down in above figure), and the direction is measured as zero degrees = column+, rotating clockwise.

If we are only interested in the magnitude, we can use the Laplacian:

$$\nabla^2 I = \frac{\partial^2 I}{\partial r^2} + \frac{\partial^2 I}{\partial c^2}$$

What does it signify?



The first derivative has a local maximum at the location of the edge, while the second derivative has a **zero-crossing** at the location of the edge. Therefore to detect an edge, we could look for either large values in the first derivative or zero-crossings in the second.

In general, first derivatives are less susceptible to corruption by noise than second (or higher) derivatives. But second derivatives allow for greater precision in defining the actual edge location.

How can we actually compute these quantities? Using convolution with appropriate filters. Probably the most common is Sobel's operator:

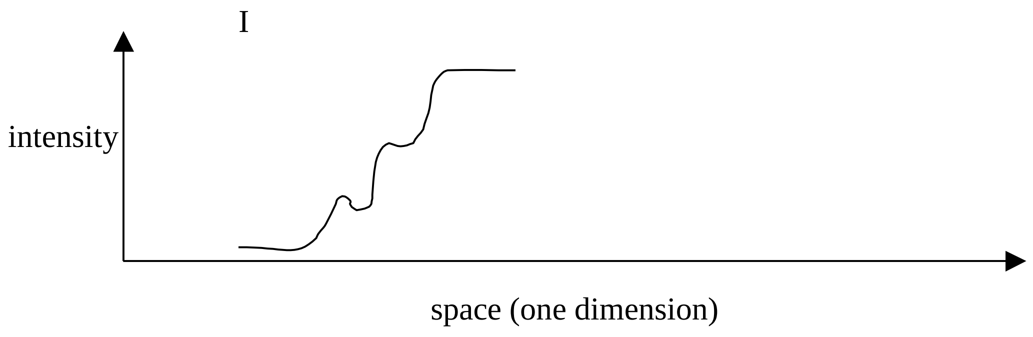
$$f_1 = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad f_2 = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

where f_1 and f_2 compute the horizontal and vertical gradients.

The Laplacian, being invariant to rotation, only requires one filter:

$$f = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

What if the edge is noisy?



We want to smooth the data first, using for example a Gaussian filter:

$$f[dr,dc] = e^{-\left(\frac{dr^2+dc^2}{2\sigma^2}\right)}$$

If we use the Laplacian operator on Gaussian-smoothed data:

$$\nabla^2\big[I[r,c] * f[dr,dc]\big]$$

it turns out these operators (Laplacian and convolution) are associative and therefore the order can be reversed:

$$= \nabla^2\big[f[dr,dc]\big] * I[r,c]$$

which takes less operations (only one convolution as opposed to two). All we have to do is design the filter. To do so, lets make the substitution (to polar coordinates)

$$x^2 = dr^2 + dc^2$$

so that

$$f[dr,dc] = f[x] = e^{-\left(\frac{x^2}{2\sigma^2}\right)}$$

Then we can compute the derivatives:

$$f'[x] = -\frac{1}{\sigma^2} x e^{-\left(\frac{x^2}{2\sigma^2}\right)}$$

$$f''[x] = \frac{1}{\sigma^2}\left(\frac{x^2}{\sigma^2} - 1\right) e^{-\left(\frac{x^2}{2\sigma^2}\right)}$$

and substitute back in for dr,dc:

$$f''[dr,dc] = K\left(\frac{dr^2 + dc^2 - \sigma^2}{\sigma^4}\right) e^{-\left(\frac{dr^2+dc^2}{2\sigma^2}\right)}$$

where K is a normalizing factor so that the sum of the elements in the filter is zero (and so they are scaled to the desired range of output). These filters are sometimes called "Mexican hats", because of their shape. The Sonka text (after Jain et. al.) shows two examples.

Looking at the Mexican hat, one can see that the outer parts of the filter are performing the smoothing, while the inner part is performing the gradient detection.

All that is left is to decide on an appropriate value for sigma. How much smoothing (before edge detection) do we want? The answer is that **smoothing and gradient detection are exact opposites**! The processes fight each other, so the balance is critical.

Sometimes the application can define the scale (Look around and identify different edges as examples.) If so, sigma can be fixed.

Another idea is to examine the change in gradient response as a function of sigma. This is called **scale-space filtering**. Imagine smoothing a one-dimensional signal I[x] using a Gaussian filter f[dx,sigma] where sigma is a variable:

$$O[x,\sigma] = I[x] * f[dx,\sigma]$$

This is like smoothing the image at all possible scales (different values of sigma). Now we can compute the second derivatives of O[x,sigma] and look for zero crossings:

$$\frac{\partial^2 O[x,\sigma]}{\partial x^2} = 0$$

In this way we find the gradient at the "best" scale of sigma (analagous to the maximum gradient at all possible values of sigma).

Obviously, with so many computations, scale-space filtering is slow. But it can be useful if nothing is known about the image data.