Image Processing : a DSP application

Dr. Rozenn DAHYOT
dahyot@mee.tcd.ie
http://www.mee.tcd.ie/˜sigmedia

1.1 Introduction

An **image** is a two-dimensional signal whose intensity at any point is a function of two spatial variables [4]. We note x and y the spatial coordinates and $f(x,y)$ the image intensity values.

Examples: photographs, still video images, radar images, etc.

An **image sequence** (e.g. television) is three-dimensional signal since the image intensity at any point is a function of the two spatial ones and the time.

Each picture element or **pixel** represents a certain physical quantity. For instance, a photograph represents the luminances of various objects as seen by the camera. But an infrared image collects the temperature profile values of a scene.

Image enhancement techniques emphasize specific image features to improve the visual perception of an image.

Examples: contrast enhancement, edge detection, sharpening, zooming, etc.

Edge detection is a problem of fundamental importance in image analysis. In typical images, edges characterize object boundaries and are therefore useful for segmentation, registration, and identification of objects in a scene. Localization, detection, and extraction of objects or details of a given type on the complex image have been long recognized as some of the most challenging problems in the field of image processing. Edge-based segmentation is one of the earliest segmentation approaches and still remains very important. It relies on edges found in an image; these edges mark locations of discontinuities in gray level, colour, texture, etc. The final aim is to group local edges into a contour picture where only edge chains with a correspondence to existing objects or image parts are present. The picture obtained utilizes one of the fundamental image attributes: coherency of the contour elements.

An **edge** is defined as a local variation of the image intensity $f(x,y)$. An edge is a jump in intensity. Several type of edge profiles are shown in figure 1.1. The figure 1.2 shows two examples of pictures, one from a cartoon with noise-free ideal edge, and the other is a photograph.

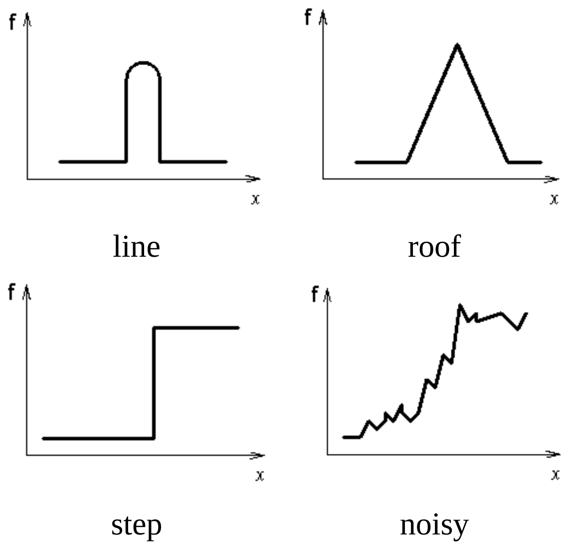


Figure 1.1: Example of edges.

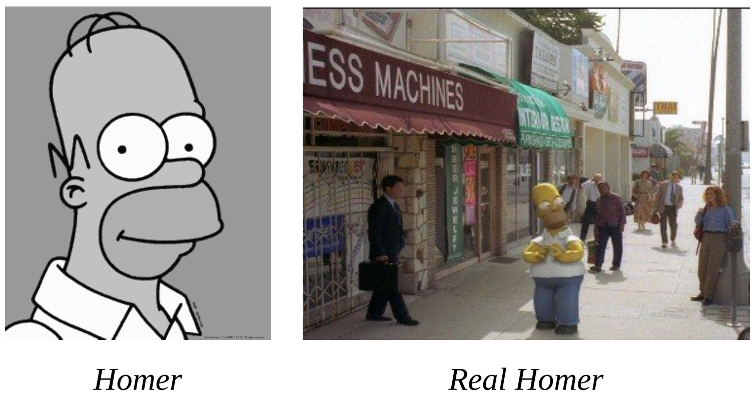


Figure 1.2: Examples of images $f(x,y)$: ideal noise-free cartoon and real picture.

1.2 Sharpening by differentiation

As shown in figure 1.3, derivatives over an edge show significant responses that are used for edge detection: (b) maxima of the first derivative and (c) zero crossings in the second derivative.

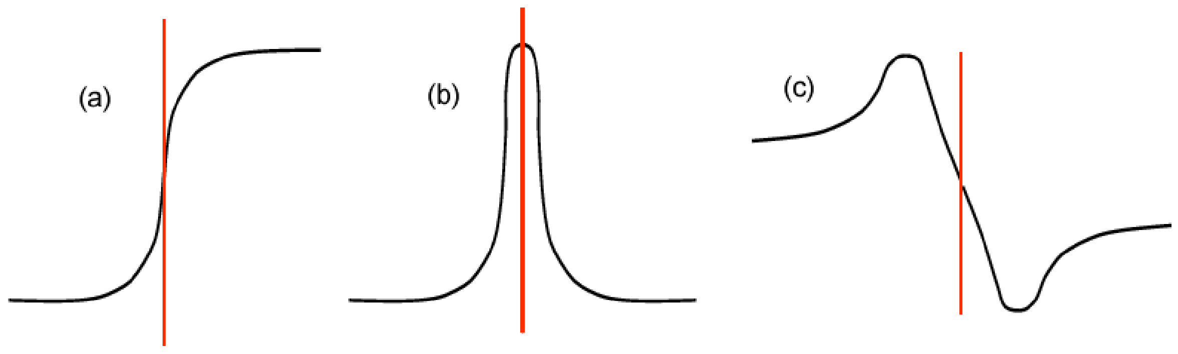


Figure 1.3: (a) 1-D signal $f(x)$, (b) its first derivative $f_x(x)$ and (c) the second one $f_{xx}(x)$.

1.2.1 First derivatives

For an image $f(x,y)$, where x and y are the row and column coordinates respectively, we typically consider the two directional derivatives $f_x(x,y)$ and $f_y(x,y)$:

$$\nabla f(x,y) = \left| \begin{array}{l} f_x(x,y) = \frac{\partial f(x,y)}{\partial x} \\ \\ f_y(x,y) = \frac{\partial f(x,y)}{\partial y} \end{array} \right.$$

The gradient magnitude is defined as :

$$|\nabla f(x,y)| = \sqrt{(f_x(x,y))^2 + (f_y(x,y))^2}$$

Figure 1.4 shows the spatial derivatives of *Homer*.

1.2.2 Second derivatives

When the first derivative achieves a maximum, the second derivative is zero. For this reason, an alternative edge-detection strategy is to locate zeros of the second derivatives. The zeros of the Laplacian are used to locate edges:

$$\Delta f(x,y) = f_{xx}(x,y) + f_{yy}(x,y) \tag{1.1}$$

1.3 Differentiation using convolution

We consider now that the variables (x,y) are discrete. The derivatives of an image (e.g. figure 1.4) can be computed by convoluting the image with specific filters. We consider the derivations

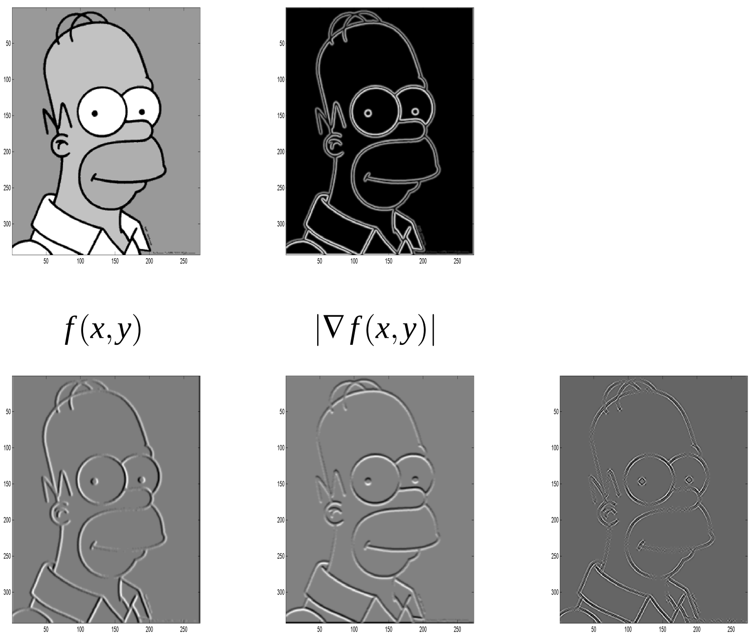


Figure 1.4: Derivatives of the image *Homer*.

in the spatial direction x and y. We note H_x (resp. H_y) the spatial filter used to compute the first derivative in the x direction (resp. y direction).

$$f_x = f \otimes H_x$$
$$f_y = f \otimes H_y$$

1.3.1 Gradient and Laplacian Masks

A standard approximation to compute the derivatives is:

$$\left| \begin{array}{l} f_x(x,y) \approx f(x,y) - f(x-1,y) \\ \\ f_y(x,y) \approx f(x,y) - f(x,y-1) \end{array} \right.$$

Then the filters H_x and H_y (called *Pixel Difference*) can be designed as $H_x = [1\ -1]$ and $H_y = [1\ -1]^T$. Or using a 3×3 matrices :

$$H_x = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix} \text{ and } H_y = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

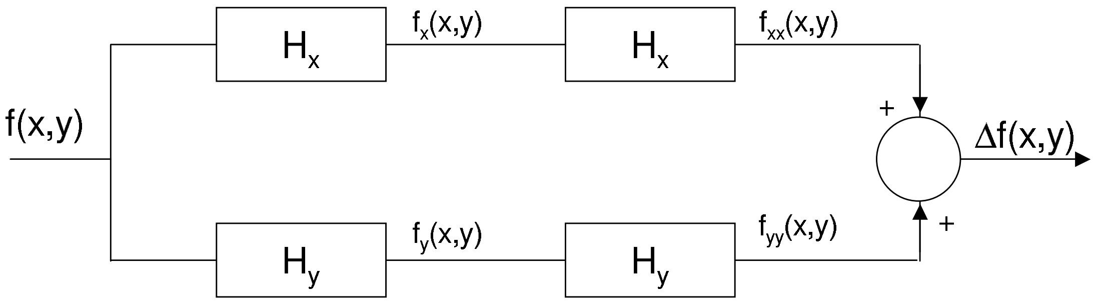


Figure 1.5: Computation of the Laplacian using only 1st derivative filters.

Those primitive filters can be used to computed the Laplacian of an image (cf. fig. 1.5).

The laplacian can be approximated by:

$$\Delta f(x,y) \approx f(x+1,y) - f(x-1,y) + f(x,y+1) - f(x,y-1) - 4f(x,y)$$

Hence the laplacian of an image can be computed directly by convolution with the following filter:

$$L = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Depending of how the derivation is approximated, other filters H_x, H_y (Sobel, Prewitt, etc.) and L can be defined.

1.3.2 Derivatives and smoothing

The previous methods are sensitive to the noise. In fact, edge filters are **high pass-filters** that enhance spatial high frequencies such as edges but also noise. One solution is to decrease the noise in the image by first applying a smoothing filter.

1.3.2.1 Smoothing using a gaussian filter

A smoother instance $\tilde{f}$ of the image f is obtained by convolution with the filter G:

$$\begin{aligned} G(x,y) &= \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2}{2\sigma^2}\right) \cdot \exp\left(-\frac{y^2}{2\sigma^2}\right) \\ &= G(x) \times G(y) \end{aligned}$$

G is a **low pass-filter** that removes very high frequencies due to noise. It is also **separable** since it can be decomposed in a product of two filters, one depending only on the variable x and the other depending only on y.

1.3.2.2 Derivative of Gaussian (DroG)

The derivative can be computed:

$$\nabla(\tilde{f} = G \otimes f) = \left| \begin{array}{l} \tilde{f}_x = G_x(x) \otimes G(y) \otimes f \\ \tilde{f}_y = G(x) \otimes G_y(y) \otimes f \end{array} \right.$$

Computing $\tilde{f}_x$ corresponds to smooth in the y direction and to a derivation in the x direction. Computing $\tilde{f}_y$ corresponds to smooth in the x direction and a derivation in the y direction. These filtering processes are then well-suited to enhanced edges oriented in the x and y directions.

1.3.3 Laplacian of Gaussian (LoG)

The Laplacian of $\tilde{f}$ can be computed:

$$\Delta(f \otimes G(x,y)) = f \otimes \Delta(G(x,y))$$

with

$$\Delta(G(x,y)) = \frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right] \exp\left(-\frac{x^2}{2\sigma^2}\right) \cdot \exp\left(-\frac{y^2}{2\sigma^2}\right)$$

$\Delta(G(x,y))$ is known as the Mexican hat filter.

1.3.4 Using Difference of Gaussian (DoG) to approximate LoG

Another method for edge detection is to apply difference of gaussian to an image. Two gaussian filters with different values of σ are applied in parallel to the image. Then the difference of the two smoothed instances $\tilde{f}_1$ and $\tilde{f}_2$ is computed. It can be shown that the DOG operator approximates the LOG one. Human visual system uses a similar method to detect edges.

1.3.5 Some Matlab Code

```
clear all ; close all;
% standard deviation of the gaussian filter
std=1;
% size of the convolution windows
x=[-5*std:5*std];
y=[-5*std:5*std]';

% filter: smoothing gaussian in the x direction
```

```
XG= exp(-x.^2/(2*std^2)) / (std*sqrt(2*pi));

% filter: smoothing gaussian in the x direction
YG= exp(-y.^2/(2*std^2)) / (std*sqrt(2*pi));

% filter: derivative gaussian in the x direction
XGx= -x.* exp(-x.^2/(2*std^2)) / (std^3*sqrt(2*pi));

% filter: derivative  gaussian in the x direction
YGy=-y.* exp(-y.^2/(2*std^2)) / (std^3*sqrt(2*pi));

% filter: laplacian Mexican hat filter
toto=(ones(length(x),1)*x.^2+ y.^2*ones(1,length(y)))/(std^2);
LoG=  (toto-2).*exp(-toto/2)/ (2*pi*std^4);

% reading an image
I = imread('homer.png');
% Ix : DroG
Ix=conv2(I,XGx,'same');
Ix=conv2(Ix,YG,'same');
% Iy : DroG
Iy=conv2(I,YGy,'same');
Iy=conv2(Iy,XG,'same');
% LoG
LoG_I=conv2(I,LoG,'same');
```

1.4 From the gradient or laplacian to the contours

The edges can be segmented using the gradient or the laplacian of an image [2]. The edge contour can be extracted in three steps:

1. computing the magnitude of the gradient $|\nabla f(x,y)|$,

2. selecting the maxima of $|\nabla f(x,y)|$ by thresholding:

$$(x,y) \in \text{contour if } |\nabla f(x,y)| > s$$

When the magnitude changes a lot in the image due to noise, this technique is inefficient since no threshold can be set to find the thin true contours.

3. To obtain thin contours, two solutions have been proposed:

   (a) Only **local extrema of the gradient** are kept. This is performed by selecting the pixel of highest magnitude of the gradient along its direction.

   (b) The detection of the contours is performed by locating the **zero-crossings** in the laplacian.

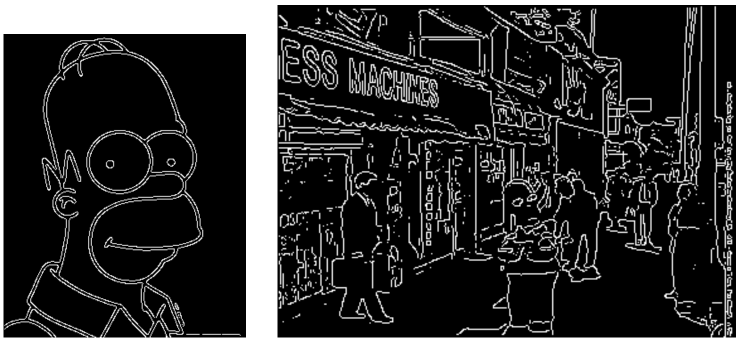


Figure 1.6: Contour segmentation on *Homer* and *Real Homer*.

1.5 Applications

1.5.1 Sport Video Indexing

The emergence of multimedia technology coupled with the rapidly expanding data collection, for private, industrial and public uses (e.g. self-made photos and videos repositories, Web resources, etc.), have attracted significant research efforts in providing tools for effective retrieval and management of the multimedia information. At first, retrieval techniques were mainly performed using keyword index. This indexing technique, mainly done manually, has since shown its limitation, both for its accuracy and for its applicability on huge database such as the world-wide web where the daily growing data cannot be handled manually. In particular, motivated by the commercial value of certain sports, retrieval and summarisation of sport events have received increasing interest in recent years. Basic image processing techniques can help to extract automatically relevant information from multimedia database.

Figure 1.7 shows results of three basic images processing tools for parsing snooker videos. Since the main information relevant to the game happens when the snooker table is appearing, a colour segmentation is applied to detect its occurrence in the images. Then edge detection is performed to extract the contours of the table. At last, a *Hough* transform is computed allowing to select only straight edges.

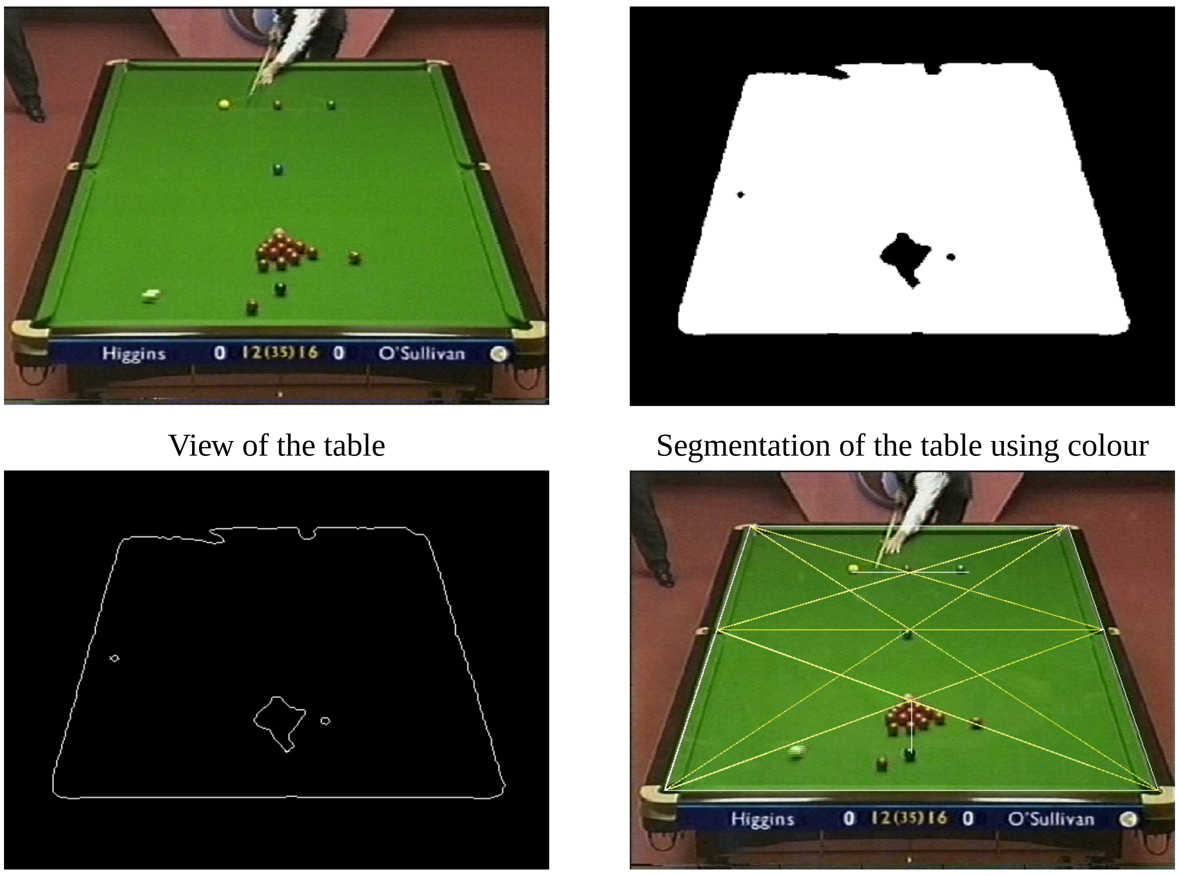

Higgins 0 12(35)16 0 O'Sullivan
View of the table
Segmentation of the table using colour
Finding the edge
Higgins 0 12(35)16 0 O'Sullivan
Finding the straight lines (using *Hough* transform)

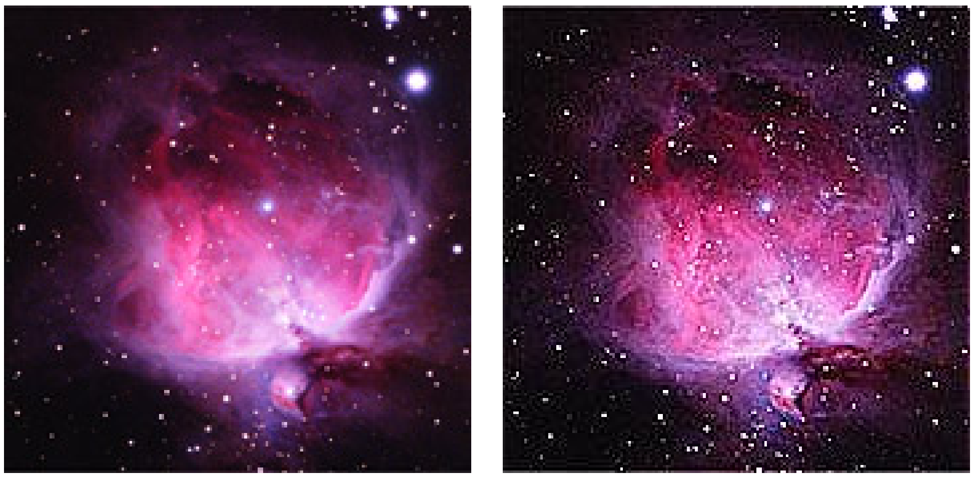
Figure 1.7: Results of segmentation of snooker table using both colour and edge information (Work performed by Niall Rea and Hugh Denman, ©Sigmedia Team).

1.5.2 Image enhancement/restoration

A well-known technique from photography to improve the visual quality of an image is to enhance the edges of the image. The technique is called **Unsharp masking**. Edge enhancement means first isolating the edges in an image, amplifying them, and then adding them back into the image. This leads immediately to the technique:

$$\hat{f}(x,y) = f(x,y) - (k \cdot \Delta f(x,y)) \tag{1.2}$$

The term k is the amplifying term and $k > 0$. Figure 1.8 shows a result of this technique on an astronomical image.



Original image Filtered image ($k = 0.2$)

Figure 1.8: Result of unsharp masking on an astronomical image.

Bibliography

[1] D. Ballard and C. Brown. *Computer Vision*. Prentice Hall- Inc. Englewood Cliffs, New Jersey, 1982. Online at: http://www.dai.ed.ac.uk/homes/rbf/BANDB/.

[2] R. Horaud and O. Monga. *Vision par ordinateur- Outils fondamentaux*. 2nd Editions Hermès, 1995. Online at: http://www.inrialpes.fr/movi/people/Horaud/livre-hermes.html.

[3] Anil Kokaram. Signals and systems handouts 3c1. Technical report, Trinity College, Dublin, Ireland, 2003.

[4] S. K. Mitra. *Digital Signal Processing- A Computer Based Approach*. Mc Graw-Hill Electrical Engineering Series, 2001.