Image filtering : Fundamentals

Silvio MONTRESOR

07.03.2023

Table des matières

I.Presentation

Module:

Imagery

Author(s):

Silvio MONTRESOR - Le Mans Université - Laboratoire d'acoustique (UMR CNRS 6613)

Abstract:

Image attributes. Discrete bidimensional signals and systems. Spatial filtering. Sampling and quantization. Digital image filters.

Keywords:

Image, Digital filtering, 2D Fourier Transform, Quantization, Sampling,

Prerequisites:

Traitement numérique du signal

Learning outcomes:

Acquérir les bases du filtrage numérique des images

Course overview:

- Introduction
- Image acquisition, representation and filtering.
- Bidimensional signals
- Image analysis in the frequency domain.
- Spatial filtering
- Sampling and quantization
- Discrete bidimensional systems
- Conclusion

Conception & production :

Le Mans Université

License:

1 - http://www.gnu.org/licenses/fdl.txt

II.Lesson

This lesson introduces basic notions related to image processing, i.e. image representation and analysis. Various aspects of image processing are represented in the bloc diagram of figure 1, which will constitute our guiding principle.

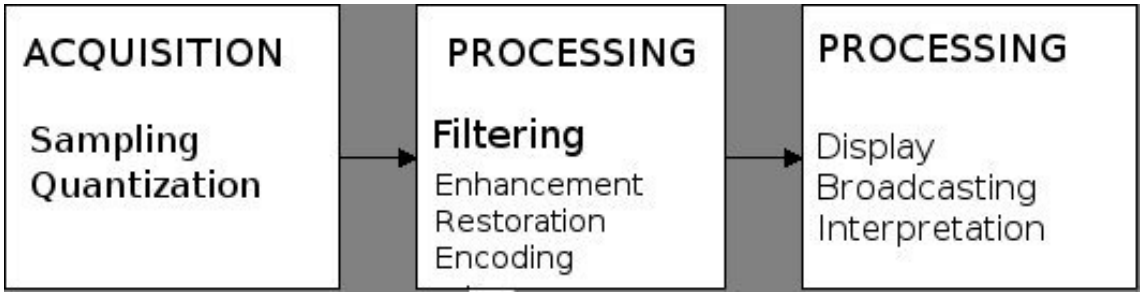


Figure 1: General diagram of the image processing chain.

More precisely, the lesson will be decomposed in five parts corresponding to the first two blocks of the previous diagram. The third and last block ("utilization") includes high-level processing such as edge detection and will not be treated in this lesson.

Concerning the acquisition part, and before any further consideration, we will present the diverse representations of an image considered as a bi-dimensional signal. A few examples of basic bi-dimensional signals and their notations will be described. Since image representation in the frequency space is of importance both for low-level and for more sophisticated processing, we will describe the properties of two-dimensional Fourier Transform (FT) in the second part of the lesson. In a third part, the FT properties will be used to introduce some important elements of image filtering that will be present in almost every low-level processing techniques, also called image preprocessing. The problem of image sampling and quantization will be presented in the fourth part. Finally, digital image filtering will be described in the last part of the lesson.

1. Image acquisition, representation and filtering

We first list the different types of images that we may have to process. In order to be processed by a computing system, an image must be represented by a discrete ensemble of values, the pixels. Each pixel is associated with one or several values depending on the chosen representation.

The most elementary representation corresponds to the binary image, for which each pixel can take only one value among two. For monochrome images, each pixel can take one value among N. N is generally a power of 2, thus facilitating the image representation in the computer. For example, for a gray-level image, each pixel can take one value among 256; this value is then encoded by a data byte. This representation is frequently used and finds a justification involving the human visual system and the physical characteristics of the image support. This point is detailed in the **Sampling and quantization section**.

A *tri-chrome* image (or color image) is a superposition of three gray-level images corresponding to three basic colors. For images that will be displayed on a computer screen, the RGB (red, green, blue) representation is used. Each pixel of a tri-chrome image is thus associated with a triplet of values corresponding to the luminance of the basic colors.

This representation is not the only one allowing color images to be processed by a computing system. An alternative consists in using indexed color images, associating the matrix of pixels with a color table (colormap). More precisely, in this case, each pixel value is an index pointing to the color table. The color table is composed of three columns corresponding to the three

basic colors. The number of rows in the table is equal to the total number of colors used for the representation. The indexed color representation is more cost-effective in terms of memory occupancy than the RGB representation since the number of colors is voluntarily limited. Therefore, it is also less precise in terms of image definition.

Images obtained through snapshots in the visible range are not the only ones to find applications, and multi-spectral images are a generalization of the previous case. They are represented by n tables of numbers.

Let us end this section by mentioning some quantitative information regarding image representation in the computer. An image is composed of a pixel matrix of N rows by M columns, where each pixel value is encoded in x bits.

1 bit	2 colors (black and white)
4 bits	16 colors
8 bits	256 colors or grey-levels
16 bits	65536 colors
24 bits	16777216 colors (true colors)
32 bits	4 294 967 296 colors

Table

For example, a tri-chrome image 512x512 for which each color is encoded in 8 bits requires 786 Ko of data for storage. Such an image is represented on figure 2. On figure 3, we represented the same image after applying the JPEG algorithm. This later image is stored in a memory space of only 15 Ko.

Figure 2 : Image "lena" tri-chrome RGB.

Figure 3 : Image "lena" obtained after JPEG compression.

This example illustrates the significant improvements that have been made in the area of digital image representation within the past fifteen years. These improvements have been applied to the video domain and have lead to new encoding norms, such as mpeg2 and mpeg4 for the most famous ones, which are widely used for digital television broadcasting.

2. Bidimensional signals

We first consider the case of continuous representations. An image can be represented with a function $f(x,y)$ associating a light intensity to a point (x,y) of the plane. To study the properties of image processing systems, we use some elementary bidimensional functions introduced here. The first function we consider is the bi-dimensional delta (or dirac) function, which is in fact a distribution and a simple extension of the monodimensional case.

$\delta(x,y) = 0 \quad \forall (x,y) \neq (0,0)$

$\delta(x,y) = +\infty \quad \text{for } (x,y) = (0,0)$

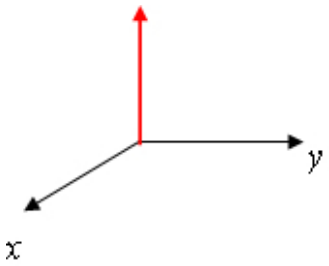


Figure 1-1 : Representation of the bidimensional delta function

Any function $f(x,y)$ can be defined using the following equation involving a two dimensional integral and the delta function as its kernel :

$$\int_{-\infty-\infty}^{+\infty+\infty} f(x-x',y-y')\delta(x',y')dx'dy' = f(x,y) \quad (1-2)$$

Using the bidimensional delta function, we also define the bidimensional comb function which corresponds to the juxtaposition of an infinity of shifted delta functions fully covering the plane (x,y),

$$P(x,y) = \sum_k \sum_l \delta(x-k\Delta x, y-l\Delta y) \quad (1-3)$$

The parameters Δx and Δy control the density of delta functions in the plane (x,y). The bidimensional unit step function (or heavyside function) $U(x,y)$ is defined by:

$U(x,y) = 1 \quad \text{if } x \geq 0 \text{ and } y \geq 0$

$U(x,y) = 0 \quad \text{elsewhere} \quad (1-4)$

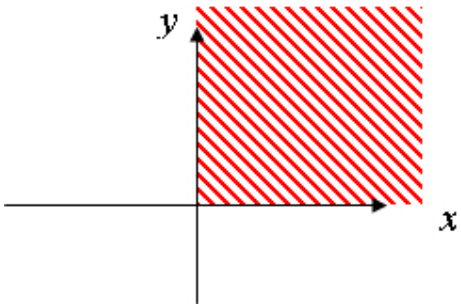


Figure 1-2 : Representation of the bidimensional unit step function;

This function covers a quarter of the plane (x,y). The bidimensional rectangular function is noted rect(x,y) and defines a region of the plane (x,y) limited by a quadrilateral,

$$rect(x,y) = 1 \quad \text{if } -\frac{1}{2} \geq x \geq \frac{1}{2} \text{ and } -\frac{1}{2} \geq y \geq \frac{1}{2}$$

$rect(x,y) = 0 \quad \text{elsewhere}$

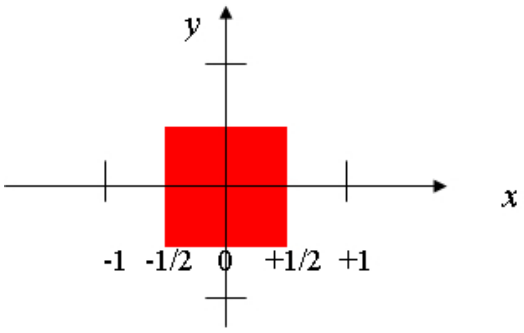


Figure 1-3 : Representation of a bidimensional rectangular function

Sinusoidal signals represent another important category of elementary signals. In particular, they are used to extract the frequency components in the bidimensional Fourier plane (see section 2). A bidimensional $S(x,y)$ sine function is characterized by its amplitude, phase, frequency and propagation direction. In the following example, the sine function is propagating along the y-axis and has a spatial frequency of .

Exemple

$S_1(x,y) = \sin(2\pi y/\lambda)$

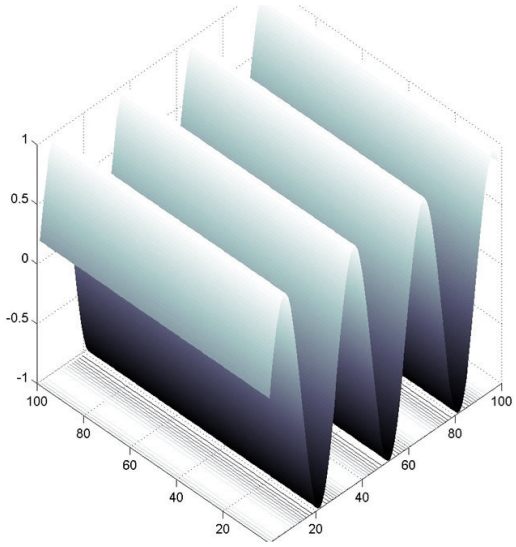


Figure 1-4 : Representation of a bidimensional sine function (example 1).

Exemple

$S_1(x,y) = \sin(2\pi(x+y)/\lambda)$

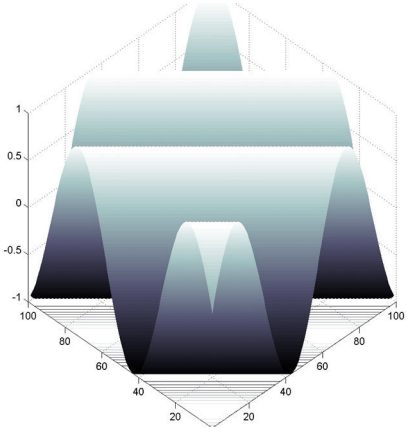


Fig : 1-5 : Representation of a bidimensional sine function (example 2).

In order to be manipulated by a computer, the analog images are transmitted to an ensemble of processes leading to their discretization. The sampling and image quantization problems will be treated in the last chapter of this section. A discrete image is represented by a matrix of points called the pixels, $I(m,n)$,

$$I(m,n) = \begin{pmatrix} x(m_1,n_1) & x(m_1,n_2) & .....x(m_1,n_N) \\ x(m_2,n_1) & & \\ \vdots & & \\ x(m_M,n_1) & & \end{pmatrix} \quad (1-5)$$

The bidimensional signal $I(m,n)$ is a real or complex function of two completely independant variables m and n. The image size represents the region of variation of the integer values defining the image.

We now present the discrete representation associated with the various analog signals defined at the beginning of this chapter. The discrete bidimensional delta function $1/\lambda$ differs from its analog counterpart in the sense that it is not a distribution but a signal equal to 1 at $(0,0)$ and 0 elsewhere. Any discrete image can be defined as a combination of shifted delta functions,

$$I(m,n) = \sum_k \sum_l I(k,l)\delta(m-k,n-l) \quad (1-6)$$

This relation also defines the bidimensional convolution product between $I(m,n)$and $\delta(m,n)$ and can be generalized by replacing $S_1(x,y) = \sin(2\pi y/\lambda)$ with a signal $h(m,n)$ representing the discrete point spread function of a discrete bidimensional filter.

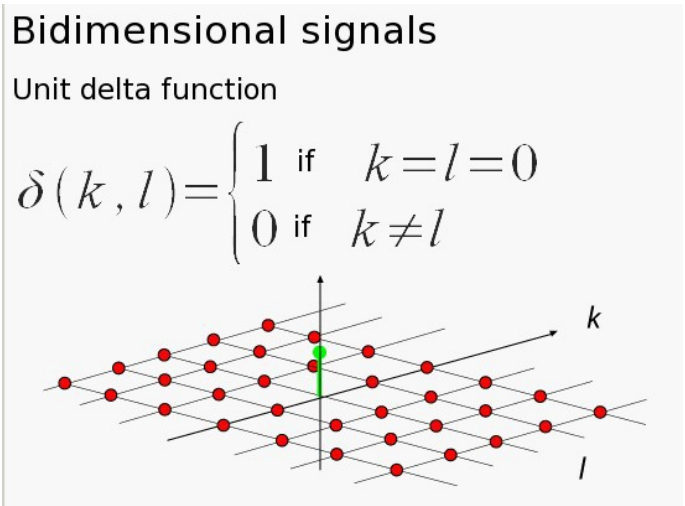


Figure 1-6 : Representation of the discrete bidimensional delta function.

The following example defines an image composed of four delta functions located next to the origin,

$$x(k,l) = \delta(k,l) + \delta(k-1,l) + \delta(k-1,l-1) + \delta(k,l-1) \quad (1-7)$$

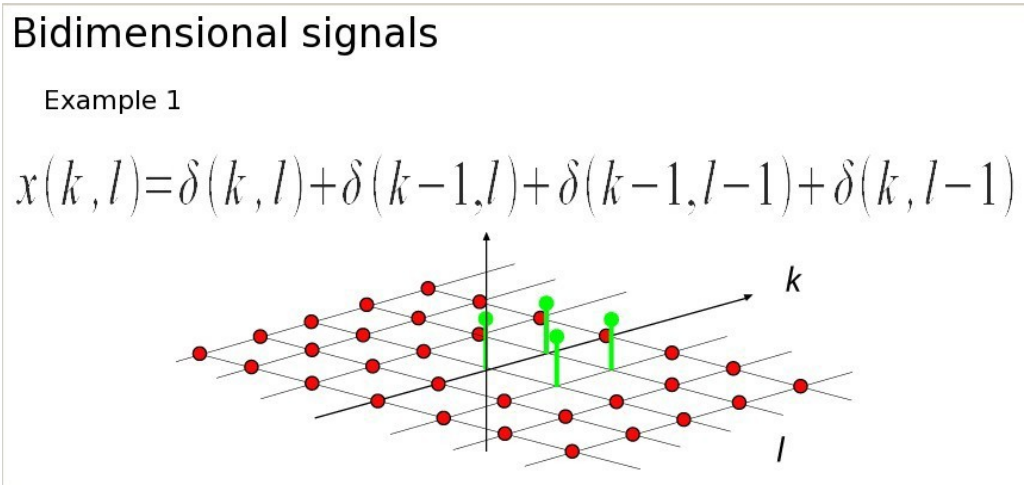


Figure 1-7 : Representation of a discrete bidimensional signal.

The discrete bidimensional unit step function $U(k,l)$ defines a region corresponding to the top right quadrant of the plane (k,l) :

$U(k,l) = 1$ if $k \geq 0$ et $l \geq 0$
$U(k,l) = 0$ elsewhere $(1-8)$

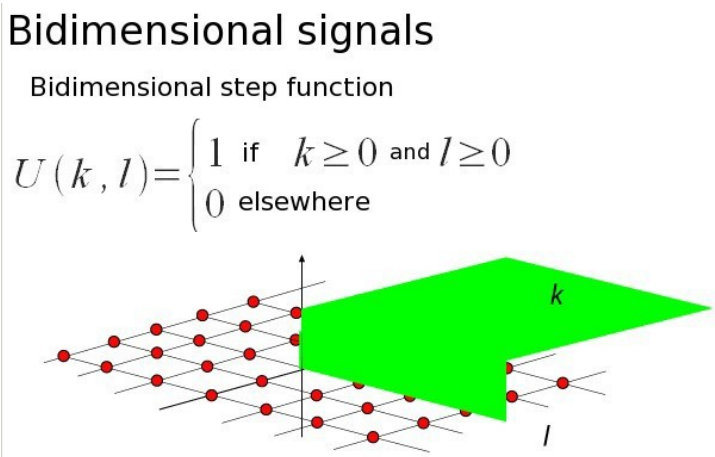


Figure 1-8 : Representation of the discrete bidimensional step function.

The discrete bidimensional rectangular function $rect_{K,L}(k,l)$ defines a rectangular region of parameters K and L. By convention, the bottom left corner coincides with the origin:

$$rect_{K,L}(k,l) = \begin{cases} 1 & \text{if } 0 \geq k \geq K-1 \text{ and } 0 \geq l \geq L-1 \\ 0 & \text{elsewhere} \end{cases} \quad (1-9)$$

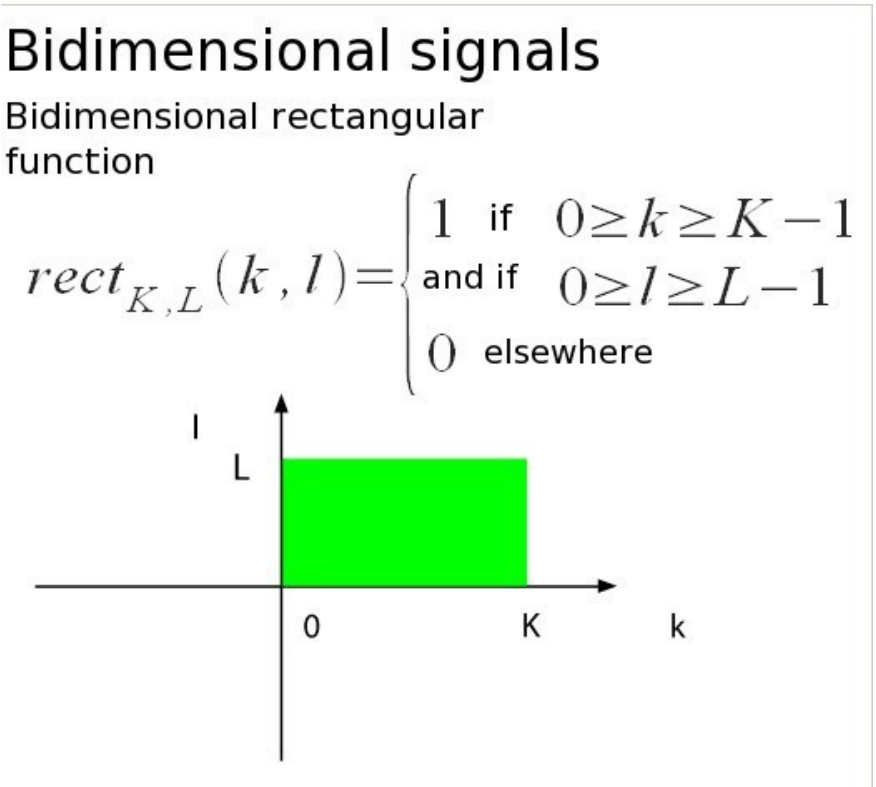


Figure 1-9 : Representation of a discrete bidimensional rectangular function.

Separable signals represent an important class of bidimensional signals. A bidimensional signal $x(k,l)$ is separable if it verifies the following property:

$$x(k,l) = x_1(k)x_2(l) \quad (1-10)$$

This property can be particularly advantageous when computing elaborate operations such as convolutions. Computations can then be treated by taking a unique dimension into account. The bidimensional delta function and step function are two examples of separable signals:

$$\delta(k,l) = \delta(k)\delta(l)$$
$$U(k,l) = U(k)U(l) \quad (1-11)$$

3. Image analysis in the frequency domain.

Classically, images are analyzed in the frequency domain using the 2D Fourier transform and its discretized and computer-optimized versions: the Discrete Fourier Transform (DFT) and most of all the Fast Fourier Transform (FFT). The programming domain and the data compression domain also resort to more adapted frequency representations such as the Discrete Cosine Transform (DCT) used for JPEG encoding.

We first consider a continuous bidimensional image $f(x,y)$.

Définition

We note $F(u,v)$ the 2D Fourier transform of image $f(x,y)$:

$$F(u,v) = \int_D \int f(x,y)e^{-2\pi j(xu+yv)}dxdy$$

D represents the region where the image $f(x,y)$ is defined. The quantities u and v are real numbers representing the spatial frequencies of the bidimensional spectrum F.

Properties

Listed below are the most important properties of the 2D Fourier Transform: spatial symmetry, separability, hermitian symmetry, spatial translation and 2D convolution (respectively).

$$\begin{array}{rcll} f(\pm x,\pm y) & \Leftrightarrow & F(\pm u,\pm v) & \\ f_1(x)f_2(y) & \Leftrightarrow & F_1(u)F_2(v) & \\ f*(x,y) & \Leftrightarrow & F(-u,-v) & (2-2) \\ f(x-x_0,y-y_0) & \Leftrightarrow & e^{-2\pi j(x_0u+y_0v)}F(u,v) & \\ f(x,y)**h(x,y) & \Leftrightarrow & F(u,v)H(u,v) & \end{array}$$

Remarque

The separability property is of particular interest in the case of a linear filtering; $f(x,y)$ then represents the impulse response of a filter. In this case, the 2D filtering operation breaks down into two 1D filtering operations, which considerably decreases the computation time. The last property (2D convolution) is also very important for filtering, and shows the compatibility of the Fourier transform with this operation. In other words, it means that the calculation of a linearly filtered image is obtained through a simple product in the spatial frequency domain. This calculation method is really advantageous when the Fourier transforms are computed with fast algorithms (FFTs).

Demonstration : separability property.

We consider a separable image $f(x,y)$, and its 2D Fourier transform $F(u,v)$,

$$F(u,v)=\iint_D f_1(x)f_2(y)e^{-2\pi j(xu+yv)}dxdy \quad (2-3)$$

We can easily notice that the kernel of the 2D FT is also separable,

$$e^{-2\pi j(xu+yv)}=e^{-2\pi jxu}e^{-2\pi jyv} \quad (2-4)$$

Therefore:

$$F(u,v)=\int_R f_1(x)e^{-2\pi jxu}dx\int_R f_2(y)e^{-2\pi jyv}dy=F_1(u)F_2(v) \quad (2-5)$$

$F1(u)$ and $F2(v)$ are the 1D Fourier transforms of $f1(x)$ and $f2(y)$. $F(u,v)$ (respectively).

Therefore, $F(u,v)$ is separable. Below, we listed the 2D FTs of a few elementary functions: the delta function, the shifted delta function, the Gaussian aperture and the rectangular aperture (respectively).

$$\begin{array}{rcll} \delta(x,y) & \Leftrightarrow & 1 & \\ \delta(x-x_0,y-y_0) & \Leftrightarrow & exp(-2\pi j(x_0u+y_0v)) & (2-6) \\ exp(-\pi(x^2+y^2)) & \Leftrightarrow & exp(-\pi(u^2+v^2)) & \\ rect(x,y) & \Leftrightarrow & sinc(u,v) & \end{array}$$

Exemple : Calculation of the 2D FT of the function rect(x,y)

We define $rect(x,y)$ as a 2D rectangular function of unit parameters. By applying the 2D FT, we obtain:

$$F(u,v)=\int_{-1/2}^{+1/2}\int_{-1/2}^{+1/2}e^{-2\pi j(xu+yv)}dxdy \quad (2-7)$$

and by using the separability of the kernel:

$$F(u,v)=\int_{-1/2}^{+1/2}e^{-2\pi jxu}dx\int_{-1/2}^{+1/2}e^{-2\pi jyv}dy \quad (2-8)$$

After integration, and using the Euler relations, we obtain:

$$F(u,v)=\int_{-1/2}^{+1/2}e^{-2\pi jxu}dx\int_{-1/2}^{+1/2}e^{-2\pi jyv}dy \quad (2-9)$$

This product defines the separable function $sinc(u,v)$. This function presents a main peak and many secondary peaks delimited by a grating of horizontal and vertical lines corresponding to

the zeros of the sine functions in the numerator of $F(u,v)$. The maximum amplitude is equal to one and is obtained at frequency $(0,0)$.

Before closing this section, we examine the case of the bidimensional sine function. We recall examples 1 and 2 from the previous chapter, and we define $S_1(x,y) = \sin(2\pi y/\lambda)$. We compute the 2D FT of S1,

$$F(u,v) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} \sin(2\pi y/\lambda)e^{-2\pi j(xu+yv)}dxdy \quad (2-10)$$

By replacing the sine function with the corresponding complex exponential functions, we find:

$$F(u,v) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} \frac{e^{j2\pi y/\lambda} - e^{-j2\pi y/\lambda}}{2j} e^{-2\pi j(xu+yv)}dxdy \quad (2-11)$$

Regrouping the exponential functions of common argument in x and y yields the sum of two terms:

$$F(u,v) = [\delta(u, v-\frac{1}{\lambda}) - \delta(u, v+\frac{1}{\lambda})]/2j \quad (2-12)$$

$F(u,v)$ is therefore constituted of two delta functions located on the v axis, on both sides from the origin $(u=0, v=0)$.

Exemple

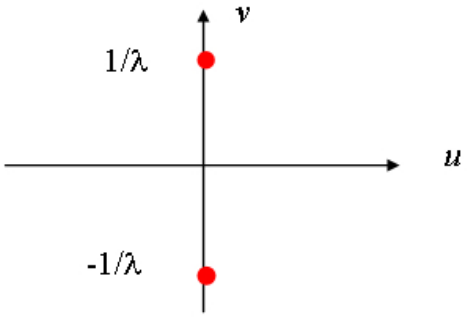


Figure 2-1 : Representation of the 2D sine function in the frequency space (example 1).

Exemple

In the case of example 2, we have $S_2(x,y) = \sin(2\pi(x+y)/\lambda)$. The 2D Fourier transform of S_2 is:

$$F(u,v) = [\delta(u-\frac{1}{\lambda}, v-\frac{1}{\lambda}) - \delta(u+\frac{1}{\lambda}, v+\frac{1}{\lambda})]/2j \quad (2-12)$$

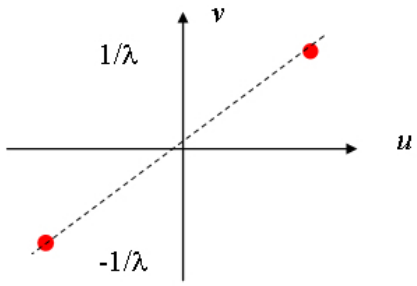


Figure 2-2 : Representation of the 2D sine function in the frequency space (example 2).

4. Spatial filtering

Filtering techniques are an important part of image processing systems, in particular when it comes to image enhancement and restoration. Here, we only consider linear and spatially invariant systems. We demonstrate that image $g(x,y)$ resulting from the passing of image $f(x,y)$ through such a system can be computed using a 2D convolution product with the system impulse response $h(x,y)$,

$$f(x,y) \rightarrow \boxed{h(x,y)} \rightarrow g(x,y) = f(x,y) * *h(x,y)$$

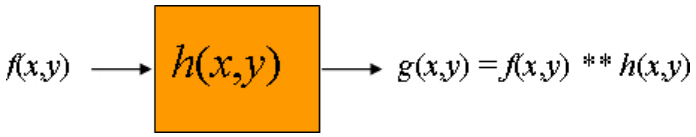


Figure 3-1 : Functional representation of a linear system invariant with spatial translation.

with:

$$g(x,y) = \int_{-\infty-\infty}^{+\infty+\infty} f(x-x', y-y')h(x',y')dx'dy'$$

In this expression, the impulse response $h(x,y)$ is defined as the system response to a bidimensional delta function,

$$h(x,y) = \int_{-\infty-\infty}^{+\infty+\infty} \delta(x-x', y-y')h(x',y')dx'dy' \quad (3-2)$$

The knowledge of this function fully characterizes the linear and spatially invariant system and allows calculating its response to any excitation. On a mathematical point of view, the bidimensional convolution product is commutative, associative, and distributive with respect to the addition. The delta function is the neutral element of this operator. One important consequence of those properties regards the combination of several linear and spatially invariant systems: the serial association of such systems is equivalent to having a unique linear and spatially invariant system whose impulse response is the convolution product of the individual impulse responses. The parallel association of several systems is equivalent to having a unique linear and invariant system whose impulse response is the sum of the individual impulse responses. In order to characterize the spectral properties of linear and spatially invariant systems, we now study the response of such a system when excited with a complex exponential function $f(x,y)$:

$$f(x,y) = e^{-2\pi(xu+yv)} \quad (3-3)$$

the quantities u and v represent the spatial frequencies along axes x and y. We now compute the response of system $h(x,y)$ to signal $f(x,y)$. By applying the convolution product to f:

$$g(x,y) = \int_{-\infty-\infty}^{+\infty+\infty} e^{+2\pi j[(x-x')u+(y-y')v]}h(x',y')dx'dy' \quad (3-4)$$

we immediately obtain:

$$g(x,y) = e^{+2\pi j(xu+yv)} \int_{-\infty-\infty}^{+\infty+\infty} e^{-2\pi j(x'u+y'v)}h(x',y')dx'dy' \quad (3-5)$$

The response is therefore the product of the initial signal $f(x,y)$ with a integral term depending only on $h(x,y)$ and on the frequencies u and v. The output image is therefore an image of the same frequencies u and v. This term is the frequency response $H(u,v)$ of the system $h(x,y)$. $H(u,v)$ exactly corresponds to the 2D Fourier transform of the impulse function $h(x,y)$.

$$H(u,v) = \int_{-\infty-\infty}^{+\infty+\infty} e^{-2\pi j(x'u+y'v)}h(x',y')dx'dy' \quad (3-6)$$

In numerous situations, determining the convolution product is facilitated by the use of the Fourier transform. Indeed, among the 2D FT properties previously listed, the one related to the convolution product directly applies to the case of linear and spatially invariant systems:

$$g(x,y) = f(x,y) * *h(x,y) \Leftrightarrow G(u,v) = F(u,v)H(u,v) \quad (3-7)$$

Computing $g(x,y)$ requires the computation of two direct Fourier transforms (applied to the image $f(x,y)$ and to the filter response) and of the reverse transform applied to the product $G(x,y)$.

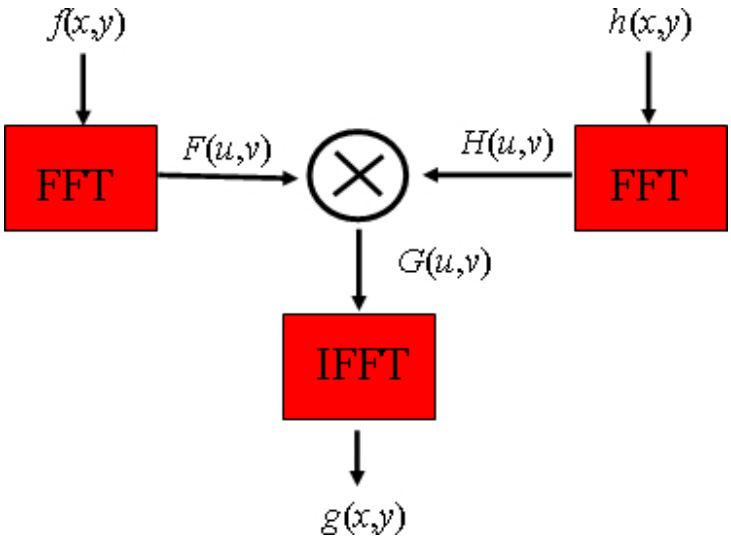


Figure -3-2 : Functional diagram of the calculation performed when spatially filtering an image using the Fourier transform.

This computation scheme is efficient if the Fourier transforms are computation-optimized: we then talk about Fast Fourier Transforms (FFT). The detailed study of the FFTs falls outside of the scope of this lesson; however, the reader can find additional information on FFTs in reference [[1] Traitement numérique du signal, théorie et pratique] .

5. Sampling and quantization

Before being processed by a computer, the analog images must be sampled and quantified. Here, we consider the ideal case of image sampling. The ideal sampler model is a simple product of the initial image with a bidimensional comb function. This calculation yields a sampled image $Ie(x,y)$ whose values correspond to the luminance registered on a regular grid of parameters $\Delta x \Delta y$.

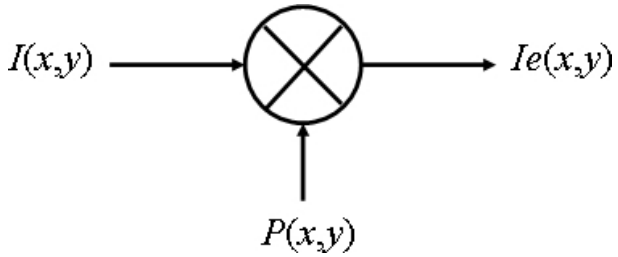


Figure 4-1 : Schematic view of an ideal sampler.

Expression of the sampled image:

$$I_e(x,y) = \sum_m \sum_n I(m\Delta x, n\Delta y)\delta(x - m\Delta x, y - n\Delta y) \quad (4-1)$$

The weights $I(mDx, nDy)$ associated with the delta functions are the pixels of the image. Shannon theorem extended to 2 dimensions is written on the same way as in the monodimensional case, except that here we find an additional degree of freedom in the

geometry of the 2D sampling pattern. However, in this lesson, we will not detail this point and we will limit ourself to a regular sampling pattern.

Définition : Bidimensional sampling theorem

An analog image $I(x,y)$ limited to the spatial frequencies $Xmax$ and $Ymax$ can be reconstructed using the sampled data $I(m\Delta x, n\Delta y)$ if the sampling periods verify:

$$\Delta x \leq \frac{1}{X_{max}} \text{ et } \Delta x \leq \frac{1}{X_{max}} \quad (4-2)$$

In order to understand the frequency representation of the sampled image, we first need to compute its Fourier Transform. We thus note:

$$Fe(u,v) = \int_{-\infty-\infty}^{+\infty+\infty} Ie(x,y) e^{-2\pi j(xu+yv)} dxdy \quad (4-3)$$

Replacing $Ie(x,y)$ by its expression yields a double integral of a double discrete sum. By linearity, we can permute the summation and integral signs to obtain:

$$Fe(u,v) = \sum_m \sum_n I(m\Delta x, n\Delta y) \int_{-\infty-\infty}^{+\infty+\infty} \delta(x - m\Delta x, y - n\Delta y) e^{-2\pi j(xu+yv)} dxdy \quad (4-4)$$

In the integral term, we recognize the Fourier transform of a delta function translated by a quantity $(m\Delta x, n\Delta y)$. By choosing unit values for $(\Delta x, \Delta y)$,we finally obtain:

$$Fe(u,v) = \sum_m \sum_n I(m,n) e^{-2\pi j(mu+nv)} \quad (4-5)$$

This is the simplified form of the bidimensional Fourier transform applied to discrete images. To understand the relation between the spectra of digital and analog images, we can notice that the image $Ie(x,y)$ is the product between the analog image and the bidimensional comb function, and therefore its Fourier transform is the convolution product of the Fourier transforms of the image $F(u,v)$ and of the comb:

$$Fe(u,v) = F(u,v) * *[\frac{1}{\Delta x \Delta y} \sum_m \sum_n \delta(u - \frac{m}{\Delta x}, v - \frac{n}{\Delta y})] \quad (4-6)$$

Therefore:

$$Fe(u,v) = \frac{1}{\Delta x \Delta y} \sum_m \sum_n F(u - \frac{m}{\Delta x}, v - \frac{n}{\Delta y}) \quad (4-7)$$

Consequently, $Fe(u,v)$ results from the periodization of the pattern formed by $F(u,v)$ along the axes u and v. $Fe(u,v)$ is a doubly periodic function of periods $(1/\Delta x, 1/\Delta y)$. It can be represented without any information loss by restricting the (spatial) frequency space to a rectangular cell of dimensions $(1/\Delta x, 1/\Delta y)$ centered for example at zero frequency $(0,0)$. The figure below shows the domain of definition of $Fe(u,v)$ restricted to this one period.

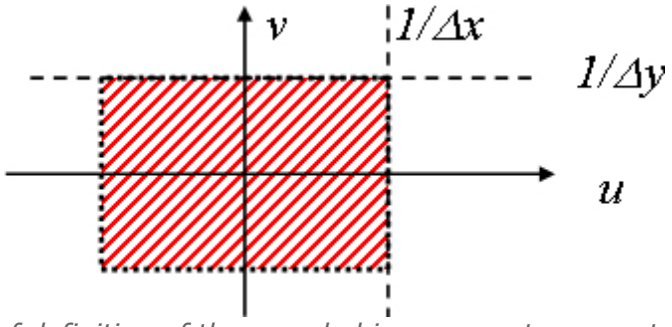


Figure 4-2: Domain of definition of the sampled image spectrum, restricted to the principal period centered at the origin.

Exemple : Calculation of the Fourier transform of a discrete rectangular function.

We consider a bidimensional discrete rectangular function $R(m,n)$ such that:

$$R(m,n) = \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} \delta(m-k, m-l) \quad (4-8)$$

By using the discrete FT expression, we obtain:

$$Fe(u,v) = \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} e^{-2\pi j(ku+lv)} \quad (4-9)$$

After separating the exponential term, we find that $Fe(u,v)$ if the product of two partial sums of well-know series. By calculating these terms, we finally obtain:

$$Fe(u,v) = e^{-j\pi(u+v)} \frac{\sin(\pi M u)}{\sin(\pi u)} \frac{\sin(\pi N v)}{\sin(\pi v)} \quad (4-10)$$

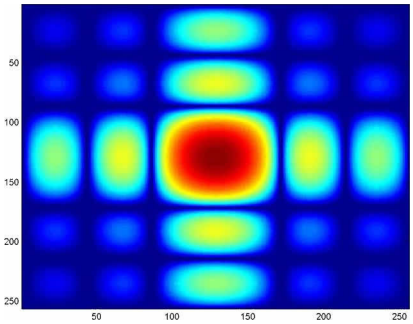


Figure 4-3: Representation of the modulus of the discrete rectangular function spectrum.

When an image is sampled without satisfying Shannon criteria, the various replica of the initial image spectrum overlap, causing the so-called aliasing phenomenon (or "moireing", in the case of images). To evidence this phenomenon, a technique consists in under-sampling an image constituted by a line grating, as shown in the following example:

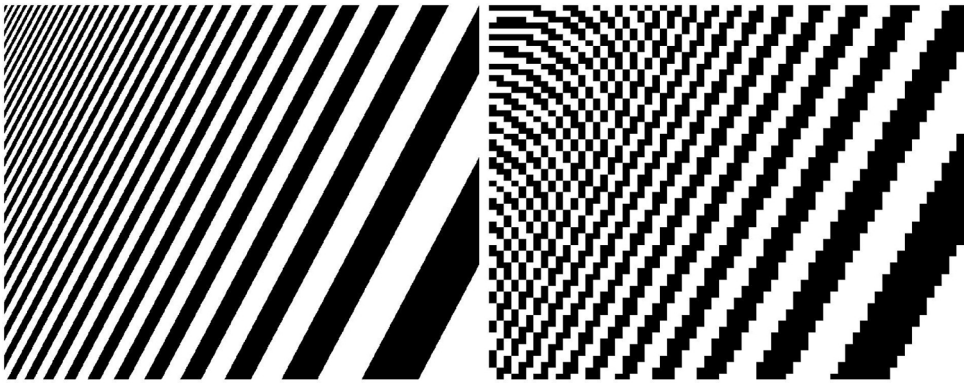


Figure 4-4 : Example

In relation to image representation in a computer, image $Ie(x,y)$ can be seen as a series of delta functions: each delta function is located at the center of a pixel and has an amplitude corresponding to the pixel intensity; the rest of the pixel has an intensity equal to zero. The pixel representation corresponds to filtering the image $Ie(x,y)$ with a rectangular filter whose impulse response is centered at the origin and has a size equal to the pixel size.

The sampling process is complemented by a quantization process, which associates a discrete value to the pixel intensity. Most of the time, the cardinal number of this ensemble of discrete values is related to the binary representation used for storage. The quantization process consists in dividing the luminance dynamic range D in a finite number of intervals. The dynamic range is determined by the physical support of the information before digitalization. The problem is then to appropriately choose the number of quantization levels. A bad choice leads to artificial outlines in the image corresponding to steps between quantization levels. The quantization interval must be chosen smaller than a quantity called luminance liminar step Cw, which represents the minimum luminance variation perceptible by a typical observer. The minimum number of levels Nq can then be expressed as a function of the dynamical range D (expressed in optical density units) and the step Cw:

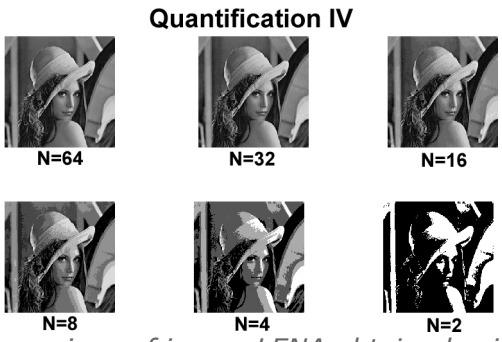
$$N_q = \frac{D}{\log 10(1 + C_w)} \quad (4-11)$$

Exemple

Using the values: $D = 2$ for a photograph and $Cw = 0.01$ for a good observer, we obtain:

$$N_q = \frac{2}{\log 10(1 + 0.01)} = 462.81$$

This leads to using $N = 512$ levels, i.e. 9 bits of quantization to code the various pixels. The same calculation performed for a medium observer $(Cw = 0.02)$ leads to $N = 256$ levels, i.e. 8 quantization bits, or a byte. This is the most frequently used value for multimedia applications.



Figures 4-5 : Various versions of image LENA obtained using the original image

6. Discrete bidimensional systems

The results regarding bidimensional filtering can be easily transposed to the domain of discrete signals. Most filters that are currently applied to these images are both linear and invariant with spatial translations. In this section, we will study in great details the mechanism of the convolution operation in the case where it is applied in the spatial domain (by opposition to the frequency domain).

Définition

A discrete bidimensional system S is defined by an operator which establishes a correspondence between two sets of bidimensional signals. To each signal $x(m,n)$ is associated a response $y(m,n)$ function of the operator S:

$$x(m,n) \rightarrow y(m,n) = S[x(m,n)]$$

Among the discrete bidimensional systems, a particular class is formed by the systems which are both linear and invariant with spatial translations. We can demonstrate that the input-output relation for such a system is given by an discrete bidimensional convolution operation:

$$y(m,n) = \sum_k \sum_l x(k,l)h(m-k,n-l)$$
$$y(m,n) = x(m,n) * *h(m,n) \qquad (5-2)$$

In this relation, $h(m,n)$ is the discrete impulse response of the system. The knowledge of $h(m,n)$ is both necessary and sufficient to fully characterize the system. For the latter relation to make sense, it is necessary that the double sum in k and l is bounded for each signal $x(m,n)$ of bounded amplitude. This condition is realized when the series of terms $h(k,l)$ converges in absolute values. We then talk about a stable system:

$$\text{if } h(m,n) \text{ is stable} : \sum_k \sum_l |h(k,l)| < \infty \qquad (5-3)$$

The discrete convolution is not the unique method allowing computing the response of a discrete bidimensional system to an excitation. An alternative is given by the finite difference equation which links the inputs and the outputs of a linear and spatially invariant system in various points of the planar spatial domain:

$$\sum_{k=0}^{K1} \sum_{l=0}^{L1} a_{k,l} y(m-k,n-l) = \sum_{k'=0}^{K2} \sum_{l'=0}^{L2} b_{k',l'} x(m-k',n-l') \qquad (5-4)$$

In addition to the knowledge of the coefficients governing the finite difference equation, it is necessary to pre-determine the boundary conditions for the output image $y(m,n)$, i.e. the values taken by the pixels at the border of the domain, where the calculation starts. These conditions are equivalent to the initial conditions defined for monodimensional systems in the temporal domain. It is important to notice that the impulse response does not explicitly appear in the calculation of the output image. This is particularly interesting when it comes to setting up image filters whose response is defined on an unbounded domain. This leads us to distinguish two categories of image filters using the finite difference equation.

6.1. Finite impulse response filters (FIR) :

This class of filters corresponds to the particular case where the ensemble of coefficients in the left member of the finite difference equation is reduced to a unique element. The output image pixel $y(m,n)$ is therefore only a function of the input image pixels. There is no recurrence.

$$y(m,n) = \frac{1}{a_{0,0}} \sum_{k'=0}^{K2} \sum_{l'=0}^{L2} b_{k',l'} x(m-k',n-l') \qquad (5-5)$$

One can realize that the impulse response directly appears in the finite difference equation, by noticing that the previous equation is a convolution relation between $x(m,n)$ and the coefficients $b(k',l')$. Therefore:

$$h(m,n) = \frac{1}{a_{0,0}} \sum_{k'=0}^{K2} \sum_{l'=0}^{L2} b_{k',l'} \delta(k',l') \qquad (5-6)$$

6.2. Infinite impulse response filters (IIR):

This is the most general case. Pixel $y(m,n)$ is computed using the pixels of the input image but also using the neighbor pixels of the output image. There is a recurrence relation between the pixels of the output image.

$$y(m,n) = \frac{1}{a_{0,0}} \left[\sum_{k'=0}^{K2} \sum_{l'=0}^{L2} b_{k',l'} x(m-k',n-l') - \sum_{k=1}^{K1} \sum_{l=1}^{L1} a_{k,l} y(m-k,n-l) \right] \qquad (5-7)$$

In this case, determining the impulse response is not trivial, and can be done by replacing $x(m,n)$ by a delta function $d(m,n)$ in the finite difference equation, and by specifying the boundary conditions. We can now address the study of the discrete convolution mechanism by restricting ourselves to the FIR filters. In this case, the output image is equal to:

$$y(m,n) = \sum_{k=k_1}^{k_1+K-1} \left(\sum_{l=l_1}^{l_1+L-1} x(k,l)h(m-l,n-k) \right) \quad (5-8)$$

The domain of definition of the impulse response $h(m,n)$ is limited to a matrix of size (K,L). For example, with $K = L = 3$, $k1 = -1$ and $l1 = -1$, we have:

$$h(m,n) = \begin{pmatrix} h(1,-1) & h(1,0) & h(1,1) \\ h(0,-1) & h(0,0) & h(0,1) \\ h(-1,-1) & h(-1,0) & h(-1,1) \end{pmatrix} \quad (5-9)$$

To calculate the output image filtered with $h(m,n)$, we apply a mask to the input image $x(m,n)$ which extracts the region involved in the calculation of pixel $y(m,n)$, in accordance with the previous relation. The full image calculation is made by iteratively shifting the mask on all the pixels of image $x(m,n)$. When the mask is placed on regions encompassing pixels falling outside of the image, those pixels are set to zero. The following example shows how this mechanism works.

Exemple

We consider an input image $A(m,n)$ and a filter $h(m,n)$ such that,

$$A(m,n) = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \text{ and } h(m,n) = \begin{pmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{pmatrix} \quad (5-10)$$

$B(m,n)$ is the output image. The first step consists in building a mask $g(m,n)$ starting from the filter. Indeed, equation (5.8) brings into play the elements $h(m-l,n-k)$ of the filter impulse response, resulting from a reversal of $h(l,k)$ combined with a translation of parameters (m,n):

$$\text{filter } h(m,n) = \begin{pmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{pmatrix} \rightarrow \text{ mask } g(m,n) = \begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix} \quad (5-11)$$

The second step consists in implementing two nested loops on matrix A in order to cover all pixels. Operation (5.8) is computed at each iteration. The figure below shows how matrix A and mask g are positioned for the first iteration:

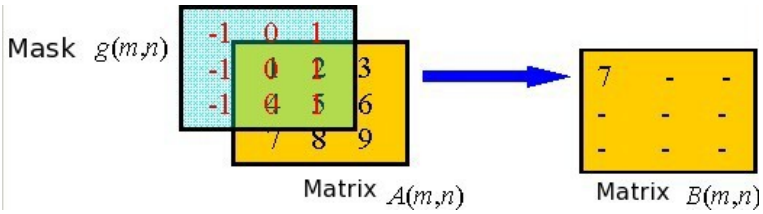


Figure 5-1: Illustration showing how image A et mask g are positioned for the filtering first iteration.

This first iteration allows calculating pixel $(1,1)$ of image B. Its value is given by:

$$B(1,1) = -1 \times 0 + 0 \times 0 + 1 \times 0 - 1 \times 0 + 0 \times 1 + 2 \times 1 - 1 \times 0 + 0 \times 4 + 1 \times 5$$

The pixels of image A can be scanned in any order. We made an arbitrary choice here, but we could have equally started by positioning the mask on the central pixel of matrix A. The full calculation gives the following matrix $B(m,n)$:

$$B(m,n) = \begin{pmatrix} 7 & 4 & -7 \\ 15 & 6 & -15 \\ 13 & 4 & -13 \end{pmatrix} \quad (5-12)$$

6.3. Examples of FIR filters

Before closing this chapter, we will review a few examples of FIR filters widely used in image processing. The following examples illustrate the results obtained when filtering a gray-level image reproducing the picture of a printed circuit board. This image mainly shows a network of horizontal and vertical lines corresponding to the current ducts.

a) Simple averaging filter (box filter)

This filter is defined by a rectangular function, which is normalized so that the gain is equal to $0dB$ at the origin of the frequency space. The impulse response of the $3x3$ filter reads:

$$h(m,n) = \frac{1}{9}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \quad (5-13)$$

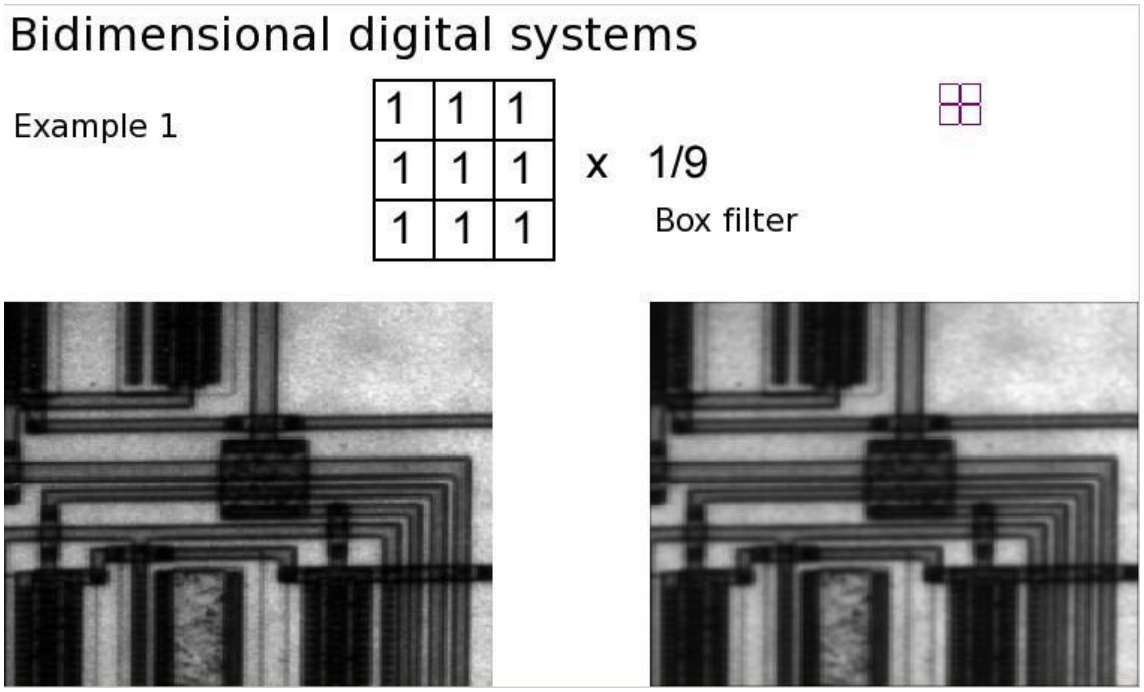


Figure 5-2 : Applying a box filter to the test image.

Such a filtering results in smoothing the discontinuities visible on the printed circuit board. The circuit lines thus look blurred. We can notice that this phenomenon is independent from the image orientation. In the frequency domain, the average filter is a low-pass filter.

b) High-pass filter

Here is the impulse response of a simple $3x3$ high-pass filter:

$$h(m,n) = \begin{pmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{pmatrix} \quad (5-14)$$

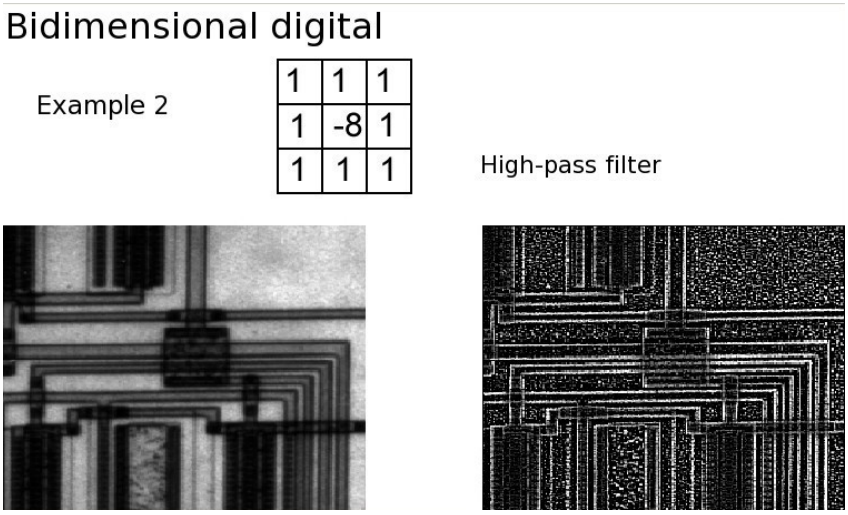


Figure 5-3: Application du filtre passe-haut sur l'image test.

This filter highlights the discontinuities of the image, and therefore we observe a strengthening of the circuit layout. We also notice a higher level of noise, which can be observed in particular in the uniform regions outside of the circuit lines.

c) The vertical differential filter

The $3x3$ vertical differential filter has an impulse response equal to:

$$h(m,n) = \begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix} \quad (5-15)$$

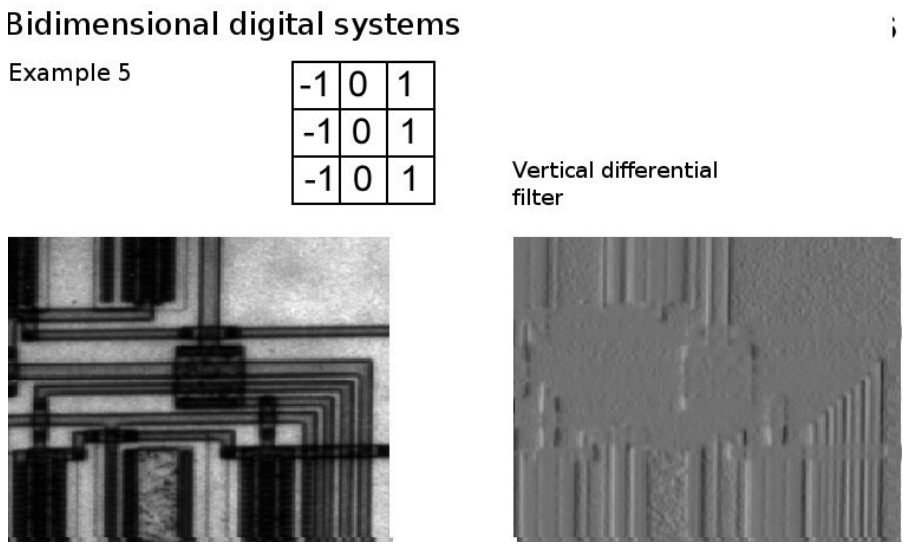


Figure 5-4: Applying the vertical differential filter to the test image

We clearly notice on the filtered image that only the contours of the vertical lines are highlighted. The horizontal lines have disappeared because their contours have been smoothed by an averaging effect on the vertical direction.

* *
*

Linear filtering is one of the most elementary and essential operation involved in image processing. Before such processing schemes can be implemented on a computer, the images have to be digitalized, which implies spatial sampling and luminance quantization. Spatial sampling induces periodicities in the frequency representation of the processed images, which allows limiting their description to a precise frequency domain. The spatial convolution operator is the mathematical representation associated with linear and spatially invariant filtering. Filtering computation can be implemented in the spatial or frequency domain depending on the nature and the complexity of the applied filters. In the simplest cases, the operator is fully described by a 2x2 or a 3x3 matrix. Examples of such filters and their results on real images are presented to give an idea of the application potential of the technique.

III.Case Study

1. Image representation and filtering

All processing examples described in the following are done using Matlab 6.1 or later, associated with the image processing toolbox 3.1 or later. Image moon.tif belongs to the image library of the toolbox.

1.1. Image formats in Matlab

Matlab can use 3 different types of data and 4 different formats to store images. The 3 types of data are:

Définition : uint8

Numbers are encoded as unsigned integers in 8 bits. Possible values range between 0 and 255. This is the most cost-effective format in terms of memory space, but it does not allow algebraic operations.

Définition : uint16

Numbers are encoded as unsigned integers in 16 bits. Possible values range between 0 and 65535. This format is similar to uint8 but offers a wider range of values. It is rarely used.

Définition : double

Numbers are encoded as double-precision floating-point values, in 64 bits. Every possible real value can be encoded, within the precision limits. This format is the less cost-effective, but allows all algebraic operations

The 4 image formats are:

Définition : indexed

The image is stored in a 2D matrix whose coefficients are indexes pointing towards an associated color table (colormap). With data types *uint8* or *uint16*, a value of 0 corresponds to the first row of the colormap, 1 corresponds to the second row, and so on. With data type *double*, a value of 1.0 corresponds to the fist row, 2 to the second, and so on.

Définition : RGB ou truecolor

The image is stored in a 3D matrix of size (m, n, 3). The third dimension gives the RGB decomposition of the pixel color.

Définition : intensity

The image is stored in a 2D matrix whose coefficients correspond to gray levels.

Définition : binary

The image is stored in a 2D matrix, which takes only 0 and 1 values (black and white, resp.). Only data types *uint8* or *double* are allowed with this image format.

Définition : colormap

A colormap is the color table associated with an indexed image. This matrix contains m rows and 3 columns of elements of data type double. Each row gives the RGB decomposition of a color. The coefficients must fall into the interval $[0:0;1:0]$.

1.2. Processing applications

1. Read and display image moon.tif in gray-levels. What is the image format? How many quantization levels are used to represent this image?

2. Modify the pixel values of image moon.tif to display it as a negative image (NB: negative image = tonal inversion of the original image, in which light areas appear dark and vice versa).

3. Filtering. We consider the following bidimensionnal separable filter $h(5,5)$:

$$b = \begin{vmatrix} 1 & 1 & 1 & 5 \\ 1 & 1 & 5 & 5 \\ 1 & 5 & 5 & 8 \\ 5 & 5 & 8 & 10 \end{vmatrix}$$

$$h(k,l) = rect_{3,3}(k+1, l+1)$$

Display the frequency response (in magnitude) of this filter. Filter image moon.tif by computing the convolution product with $h(5,5))$ and display the result.

We now consider filter

4. What conclusion can we draw from this last point?

2. Answers to the processing applications

2.1. Answers to the processing applications

1. Read and display image moon.tif in gray-levels. What is the image format? How many quantization levels are used to represent this image?

```
>>[ x, m]  = imread('moon.tif');
>> size(x)

ans =

537 358
>> size(m)

ans =

    0     0
>> imshow(x)
```

Figure C1: Display of the original image

```
>> max(max(x))

ans =

   253
>> min(min(x))

ans =

     0
```

Remarque

moon.tif corresponds to a gray-level monochrome image. The minimum and maximum values are 0 and 253. The pixels are therefore quantized on 8 bits. It is not an indexed image since the color table is empty.

2. Modify the pixel values of image moon.tif to display it as a negative image.

A simple method is to create a uniform matrix y of same dimension as x, whose pixels are equal to 255. We then compute the difference between y and x, and display the result.

```
>> y = ones(537,358)*255;
>> z = uint8(double(y)-double(x));
>> imshow(z)
```

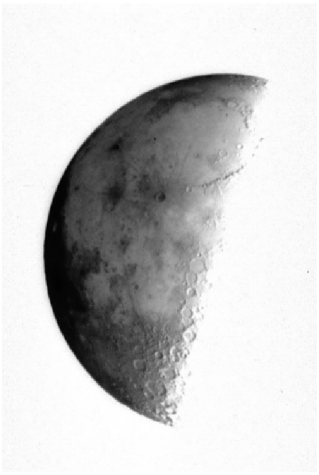


Figure C2 : display of the negative image.

Remarque

The difference can only be computed after conversion of the images from their original format uint8 to the format double. The output image then needs to be back-converted to uint8 before being displayed.

3.Filtering. We consider the following bidimensionnal separable filter h(5,5):

$$\begin{matrix}0.0357\\0.2411\\0.4464\\0.2411\\0.0357\end{matrix} \times [0.0357 \quad 0.2411 \quad 0.4464 \quad 0.2411 \quad 0.0357] =$$

$$\begin{matrix}0.00127449 & 0.00860727 & 0.01593648 & 0.00860727 & 0.00127449\\0.00860727 & 0.05812921 & 0.10762704 & 0.05812921 & 0.00860727\\0.01593648 & 0.10762704 & 0.19927296 & 0.10762704 & 0.01593648\\0.00860727 & 0.05812921 & 0.10762704 & 0.05812921 & 0.00860727\\0.00127449 & 0.00860727 & 0.01593648 & 0.00860727 & 0.00127449\end{matrix}$$

Filter creation:

```
>>  k =[0.0357 0.2411 0.4464 0.411 0.0357]

k =

   0.0357   0.2411   0.4464   0.4110   0.0357

>> h = k'*k

h =

   0.0013   0.0086   0.0159   0.0147   0.0013
   0.0086   0.0581   0.1076   0.0991   0.0086
   0.0159   0.1076   0.1993   0.1835   0.0159
   0.0147   0.0991   0.1835   0.1689   0.0147
   0.0013   0.0086   0.0159   0.0147   0.0013
```

Display the frequency response (in magnitude) of this filter.

```
>> freqz2(h)
```

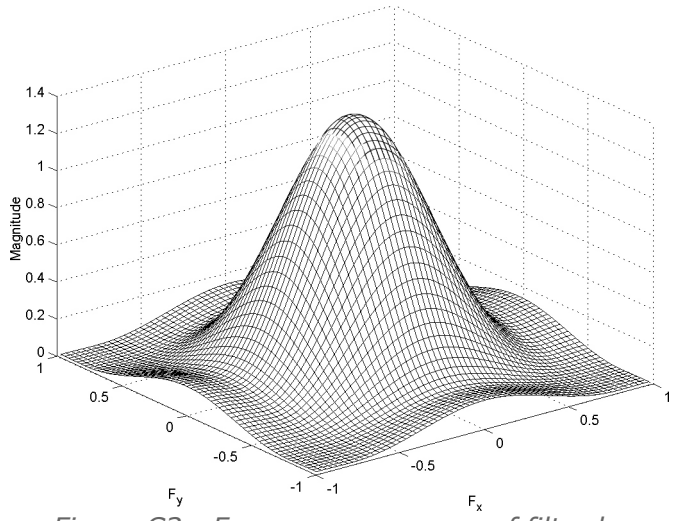


Figure C3 : Frequency response of filter h.

Remarque

The frequency response of h is maximum at zero frequency (i.e. at the frequency origin). It is therefore a low-pass filter.

Filter image moon.tif by computing the convolution product with h(5,5) and display the result.

```
>> fx = imfilter(x,h) ;

>> imshow(fx)
```

Figure C4: Image moon.tif filtered with h

We now consider the filter g:

$$\frac{1}{6}\begin{vmatrix} -1 & -4 & -1 \\ -4 & 26 & -4 \\ -1 & -4 & -1 \end{vmatrix}$$

>>g = (1/6)*[-1 -4 -1;-4 26 -4; -1 -4 -1]

g =

-0.1667	**-0.6667**	**-0.1667**
-0.6667	**4.3333**	**-0.6667**
-0.1667	**-0.6667**	**-0.1667**

Display its frequency response:

>> freqz2(g)

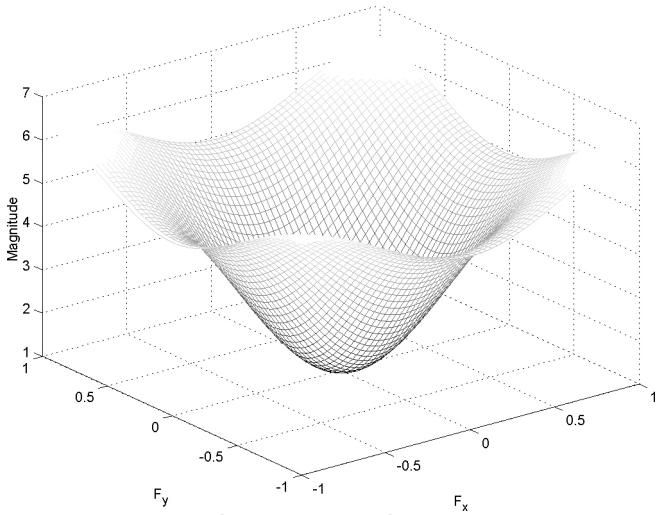


Figure C5 : Réponse en fréquence du filtre g

and apply the filter to the output image from the last step.

>> t = imfilter(fx,g);
>> imshow(t)

Figure C6 : image moon.tif filtered successively with h and g

4. What conclusion can we draw from this last point?

We observe that the details and more particularly the crater contours have been highlighted by this later filtering, while they had been attenuated by the first filtering.

Remarque

This example illustrates a technique used in image enhancement called « unsharp masking ».

IV.Exercises

1. Exercises on image filtering

1.1. Exercises on image filtering

We consider a matrix $b(4,4)$ representing samples of an image B, such that:

$$b = \begin{vmatrix} 1 & 1 & 1 & 5 \\ 1 & 1 & 5 & 5 \\ 1 & 5 & 5 & 8 \\ 5 & 5 & 8 & 10 \end{vmatrix}$$

Question 1

[Solution n°1 p 31]

Filter this image using the $3x3$ centered bidimensional rectangular filter of impulse response $h(k,l)$:

$h(k,l) = rect_{3,3}(k+1, l+1)$

We now consider a filter $h(k,l)$ of impulse response:

$$\begin{vmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{vmatrix}$$

Question 2

[Solution n°2 p 31]

Write this impulse response as function of a sum of shifted delta function.

Question 3

[Solution n°3 p 31]

Show that filter $h(k,l)$ is separable. To do so, you will determine two monodimensional filters $h1(k)$ et $h2(l)$ verifying:

$h = (h2^T) \cdot h1$

We now consider an image $a(k,l)$:

$$a = \begin{vmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{vmatrix}$$

Question 4

[Solution n°4 p 31]

Calculate $a ** h$ and $(a * h1) * (h2T)$. Conclusion?

Answer the following question.

Question 5

[Solution n°5 p 32]

We consider a film of dynamic range $D = 3.6$ (given in optical density units). What is the minimum number of bits that we must use to quantize the luminance of an image registered on this film so that the quantization error is invisible to a typical observer?

Solution des exercices

>Solution n°1 *(exercice p. 30)*

We consider the filter $h(k,l)$ defined by:
$h(k,l) = rect_{3,3}(k+1, l+1)$
This filter corresponds to the matrix:

$$\begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{vmatrix}$$

Filtering image B with filter h gives an output of:

$$b**h = \begin{vmatrix} 4 & 10 & 18 & 16 \\ 10 & 21 & 36 & 29 \\ 18 & 36 & 52 & 41 \\ 16 & 29 & 41 & 31 \end{vmatrix}$$

>Solution n°2 *(exercice p. 30)*

We consider the filter $h(k,l)$ of impulse response:

$$\begin{vmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{vmatrix}$$

This response can be expressed as a sum of shifted delta functions:
$h(k,l) = \delta(k+1, l-1) + \delta(k+1, l) + \delta(k+1, l+1) - \delta(k-1, l-1) - \delta(k-1, l) - \delta(k-1, l+1)$

>Solution n°3 *(exercice p. 30)*

$h(k,l)$ is the result of the combination of the following two vectors $h1$ et $h2$. It is therefore a separable filter.
$h = (h2^T) \cdot h1 = [1 \;\; 1 \;\; 1]^T \cdot [1 \;\; 0 \;\; -1]$
We now consider image $a(k,l)$:

$$a = \begin{vmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{vmatrix}$$

>Solution n°4 *(exercice p. 30)*

Calculation of $a**h$:

$$a**h = \begin{vmatrix} 7 & 4 & -7 \\ 15 & 6 & -15 \\ 13 & 4 & -13 \end{vmatrix}$$

Calculation of $(a * h1)$:

$$a**h1 = \begin{vmatrix} 2 & 2 & -2 \\ 5 & 2 & -5 \\ 8 & 2 & -4 \end{vmatrix}$$

Calculation of $(a * h1) * (h2T)$:

$$(a * h1) * (h2^T) = \begin{vmatrix} 7 & 4 & -7 \\ 15 & 6 & -15 \\ 13 & 4 & -13 \end{vmatrix}$$

Conclusion :

The two calculation procedures are equivalent and lead to the same result. For a separable filter, the 2D convolution product can therefore be decomposed in two 1D convolution productions.

>Solution n°5 *(exercice p. 30)*

With $D = 3.6$ optical density units and a typical observer $(Cw = 0.02)$, the number of quantization levels is equal to:

$$N_q = \frac{D}{\log 10(1 + C_w)} = \frac{3.6}{\log 10(1.02)} = 418.5963$$

By taking the closest higher power of two, we find $N = 512$, which means that the images must be quantized using 9 bits.