THE À TROUS ALGORITHM

The name *à trous* (which means "with holes") is due to the introduction of dummy elements inside successive kernels to simulate the shrinking of the image. This process can be thought of as a pyramidal structure which, in turn, highlights structures with different dimensions.

Let us assume that the input image I can be obtained as the scalar product:

$$I_0(\boldsymbol{p}) = I(\boldsymbol{p}) = \langle f(\boldsymbol{q}), \phi(\boldsymbol{q} - \boldsymbol{p}) \rangle \tag{1}$$

where ϕ is the so called *scaling function* in dyadic terms:

$$\frac{1}{4}\phi\Big(\frac{\boldsymbol{p}}{2^i}\Big) = \sum_{\boldsymbol{q} \in \mathrm{D}_\infty} l_{\boldsymbol{q}} \phi\Big(\frac{\boldsymbol{p}}{2^{i-1}} - \boldsymbol{q}\Big) \tag{2}$$

and l is a low-pass filter to be defined.
From a general point of view, the signal f is a function to represent I in terms of ϕ; in our case, f gives the intensity values of I.

Successive approximations I_i of I_0 can be computed by:

$$I_i(\boldsymbol{p}) = \frac{1}{4^i}\Big\langle f(\boldsymbol{q}), \phi\Big(\frac{\boldsymbol{q} - \boldsymbol{p}}{2^i}\Big)\Big\rangle = \sum_{\boldsymbol{q} \in \mathrm{D}_\infty} l_{\boldsymbol{q}} I_{i-1}(\boldsymbol{p} + 2^{i-1}\boldsymbol{q}). \tag{3}$$

In contrast, the wavelet coefficients W_i are obtained by convolving I_{i-1} with a high-pass filter h:

$$W_i(\boldsymbol{p}) = \frac{1}{4^i}\Big\langle f(\boldsymbol{q}), \psi\Big(\frac{\boldsymbol{q} - \boldsymbol{p}}{2^i}\Big)\Big\rangle = \sum_{\boldsymbol{q} \in \mathrm{D}_\infty} h_{\boldsymbol{q}} I_{i-1}(\boldsymbol{p} + 2^{i-1}\boldsymbol{q}) \tag{4}$$

where ψ is called *wavelet function*:

$$\frac{1}{4^i}\psi\Big(\frac{\boldsymbol{p}}{2^i}\Big) = \frac{1}{4^{i-1}} \sum_{\boldsymbol{q} \in \mathrm{D}_\infty} h_{\boldsymbol{q}} \phi\Big(\frac{\boldsymbol{p}}{2^{i-1}} - \boldsymbol{q}\Big).$$

The simplest choice for the high-pass filter h is the difference between two consecutive spatial scales:

$$W_i(\boldsymbol{p}) = I_{i-1}(\boldsymbol{p}) - I_i(\boldsymbol{p}) \tag{5}$$

that leads to (cmp. eqs. (3) and (4)):

$$\frac{1}{4}\psi\Big(\frac{\boldsymbol{p}}{2}\Big) = \phi(\boldsymbol{p}) - \frac{1}{4}\phi\Big(\frac{\boldsymbol{p}}{2}\Big).$$

To define the low-pass filter l (2), we have compared the results obtained via different wavelet functions and preferred the B_s*-spline* odd function because it is isotropic, limited and returns sharp wavelet images W_i:

$$B_s(z) = \frac{1}{2s!}\sum_{t=0}^{s+1}(-1)^t \begin{pmatrix} t \\ s+1 \end{pmatrix} \Big|z + t - \frac{s+1}{2}\Big|^s.$$

By imposing in $\phi = B_s$ the linear interpolation ($s = 1$) and the separability of the variables ($\phi(\boldsymbol{p}) = \phi(x)\phi(y)$), we get:

$$\phi(\boldsymbol{p}) = \frac{1}{4}(|x-1| - 2|x| + |x+1|)(|y-1| - 2|y| + |y+1|).$$

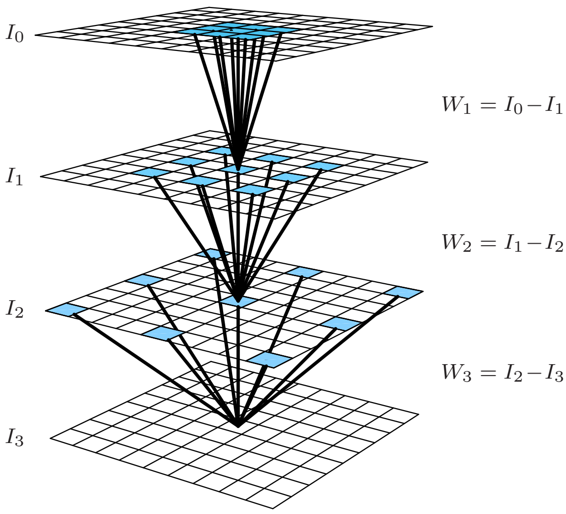

Fig. 1. Details in I_0 are highlighted by convolving with a 3×3 low-pass kernel l. Bigger structures are located by stretching l. The difference between the resulting images is a high-pass filter h that better isolates the structures with a given size.

For performance reasons, we limit D_∞ to a disk with radius $\sqrt{2}$ and compare the above polynomial with the scaling function (2), considered with $i = 0$:

$$\phi(\boldsymbol{p}) = \sum_{\boldsymbol{q} \in \mathrm{D}_{\sqrt{2}}} 4 l_{\boldsymbol{q}} \phi(2\boldsymbol{p} - \boldsymbol{q})$$

we obtain a system of 25 linear equations in 9 unknowns with solution:

$$l = \frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}.$$

Figure 2 depicts both the two-dimensional ϕ scaling function (when the B_1*-spline* has been adopted as ϕ) and its corresponding wavelet function ψ.

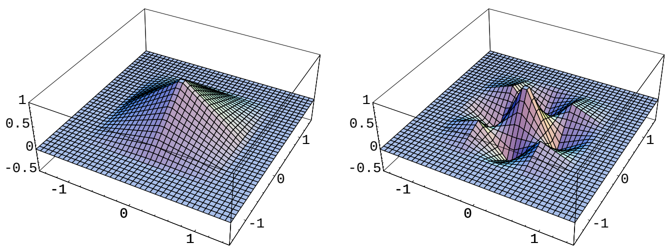

Fig. 2. Two-dimensional B_1*-spline* scaling function ϕ (*left*) and wavelet function ψ (*right*).

It must be noted that, through the separability of the variables ($h(\boldsymbol{p}) = h(x)h(y)$), the two-dimensional kernel can be simplified to the mono-dimensional version (to be applied along both the x and y axes):

$$l = \frac{1}{4}\begin{pmatrix} 1 & 2 & 1 \end{pmatrix}.$$

Actually, the *à trous* algorithm takes a constant time when computing W_i, due to the advantage that, according to eq. (3), the size of l does not change: instead, to obtain a multi-scale analysis, at each iteration i, the distance between the central pixel and adjacent ones in I_{i-1} is 2^{i-1}.

In order to compute the kernel l used by the *à trous* algorithm, let us consider the expansion of the scaling function as a linear interpolation:

$$\phi(\boldsymbol{p}) = \frac{1}{4}(|x-1|-2|x|+|x+1|)(|y-1|-2|y|+|y+1|) =$$

$$\frac{1}{4}|x-1||y-1| - \frac{1}{2}|x-1||y| + \frac{1}{4}|x-1||y+1| - \frac{1}{2}|x||y-1| + |x||y|$$

$$-\frac{1}{2}|x||y+1| + \frac{1}{4}|x+1||y-1| - \frac{1}{2}|x+1||y| + \frac{1}{4}|x+1||y+1|.$$

which can be evaluated on the 3×3 window $\mathrm{D}_{\sqrt{2}}$:

$$\phi(\boldsymbol{p}) = \sum_{x_q,y_q=-1}^{1} 4 l_{\boldsymbol{q}} \phi(2\boldsymbol{p}-\boldsymbol{q}) = |2x-1||2y-1|(l_{0,0}-2l_{0,1}-2l_{1,0}$$

$$+4l_{1,1}) + 2|2x-1||y-1|(l_{0,1}-2l_{1,1}) + 2|2x-1||y|(l_{0,-1}-2l_{0,0}+$$

$$l_{0,1}-2l_{1,-1}+4l_{1,0}-2l_{1,1}) + 2|2x-1||y+1|(l_{0,-1}-2l_{1,-1}) + |2x-1|$$

$$|2y+1|(-2l_{0,-1}+l_{0,0}+4l_{1,-1}-2l_{1,0}) + 2|x-1||2y-1|(l_{1,0}-2l_{1,1})$$

$$+4|x-1||y-1|l_{1,1} + 4|x-1||y|(l_{1,-1}-2l_{1,0}+l_{1,1}) + 4|x-1||y+1|$$

$$l_{1,-1} + 2|x-1||2y+1|(-2l_{1,-1}+l_{1,0}) + 2|x||2y-1|(l_{-1,0}-2l_{-1,1}-$$

$$2l_{0,0}+4l_{0,1}+l_{1,0}-2l_{1,1}) + 4|x||y-1|(l_{-1,1}-2l_{0,1}+l_{1,1}) + 4|x||y|$$

$$(l_{-1,-1}-2l_{-1,0}+l_{-1,1}-2l_{0,-1}+4l_{0,0}-2l_{0,1}+l_{1,-1}-2l_{1,0}+l_{1,1})+$$

$$4|x||y+1|(l_{-1,-1}-2l_{0,-1}+l_{1,-1}) + 2|x||2y+1|(-2l_{-1,-1}+l_{-1,0}+$$

$$4l_{0,-1}-2l_{0,0}-2l_{1,-1}+l_{1,0}) + 2|x+1||2y-1|(l_{-1,0}-2l_{-1,1})+$$

$$4|x+1||y-1|l_{-1,1} + 4|x+1||y|(l_{-1,-1}-2l_{-1,0}+l_{-1,1}) + 4|x+1|$$

$$|y+1|l_{-1,-1} + 2|x+1||2y+1|(-2l_{-1,-1}+l_{-1,0}) + |2x+1||2y-1|$$

$$(-2l_{-1,0}+4l_{-1,1}+l_{0,0}-2l_{0,1}) + 2|2x+1||y-1|(-2l_{-1,1}+l_{0,1})+$$

$$2|2x+1||y|(-2l_{-1,-1}+4l_{-1,0}-2l_{-1,1}+l_{0,-1}-2l_{0,0}+l_{0,1}) + 2$$

$$|2x+1||y+1|(-2l_{-1,-1}+l_{0,-1}) + |2x+1||2y+1|(4l_{-1,-1}-2l_{-1,0}$$

$$-2l_{0,-1}+l_{0,0}).$$

The comparison of these two polynomials, which must be valid for any pixel $\boldsymbol{p}$, leads to the following system:

$$\begin{cases}
l_{0,0}-2l_{0,1}-2l_{1,0}+4l_{1,1}=0\\
l_{0,1}-2l_{1,1}=0\\
l_{0,-1}-2l_{0,0}+l_{0,1}-2l_{1,-1}+4l_{1,0}-2l_{1,1}=0\\
l_{0,-1}-2l_{1,-1}=0\\
2l_{0,-1}-l_{0,0}-4l_{1,-1}+2l_{1,0}=0\\
l_{1,0}-2l_{1,1}=0\\
16l_{1,1}-1=0\\
8(l_{1,-1}-2l_{1,0}+l_{1,1})+1=0\\
16l_{1,-1}-1=0\\
2l_{1,-1}-l_{1,0}=0\\
l_{-1,0}-2l_{-1,1}-2l_{0,0}+4l_{0,1}+l_{1,0}-2l_{1,1}=0\\
8(l_{-1,1}-2l_{0,1}+l_{1,1})+1=0\\
4(l_{-1,-1}-2l_{-1,0}+l_{-1,1}-2l_{0,-1}+4l_{0,0}-2l_{0,1}+l_{1,-1}-2l_{1,0}+l_{1,1})-1=0\\
8(l_{-1,-1}-2l_{0,-1}+l_{1,-1})+1=0\\
2l_{-1,-1}-l_{-1,0}-4l_{0,-1}+2l_{0,0}+2l_{1,-1}-l_{1,0}=0\\
l_{-1,0}-2l_{-1,1}=0\\
16l_{-1,1}-1=0\\
8(l_{-1,-1}-2l_{-1,0}+l_{-1,1})+1=0\\
16l_{-1,-1}-1=0\\
2l_{-1,-1}-l_{-1,0}=0\\
2l_{-1,0}-4l_{-1,1}-l_{0,0}+2l_{0,1}=0\\
2l_{-1,1}-l_{0,1}=0\\
2l_{-1,-1}-4l_{-1,0}+2l_{-1,1}-l_{0,-1}+2l_{0,0}-l_{0,1}=0\\
2l_{-1,-1}-l_{0,-1}=0\\
4l_{-1,-1}-2l_{-1,0}-2l_{0,-1}+l_{0,0}=0
\end{cases}$$

that admits:

$$l_{-1,-1} = l_{-1,1} = l_{1,-1} = l_{1,1} = \frac{1}{16},$$
$$l_{-1,0} = l_{0,-1} = l_{0,1} = l_{1,0} = \frac{1}{8},$$
$$l_{0,0} = \frac{1}{4}.$$