

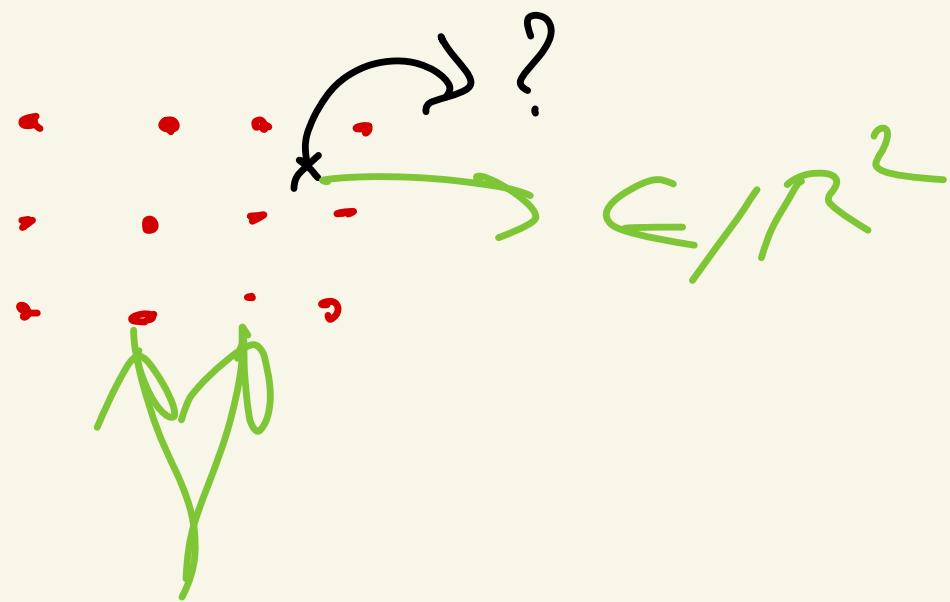
Interpolation et transformations géométriques.

Filtrage et débruitage

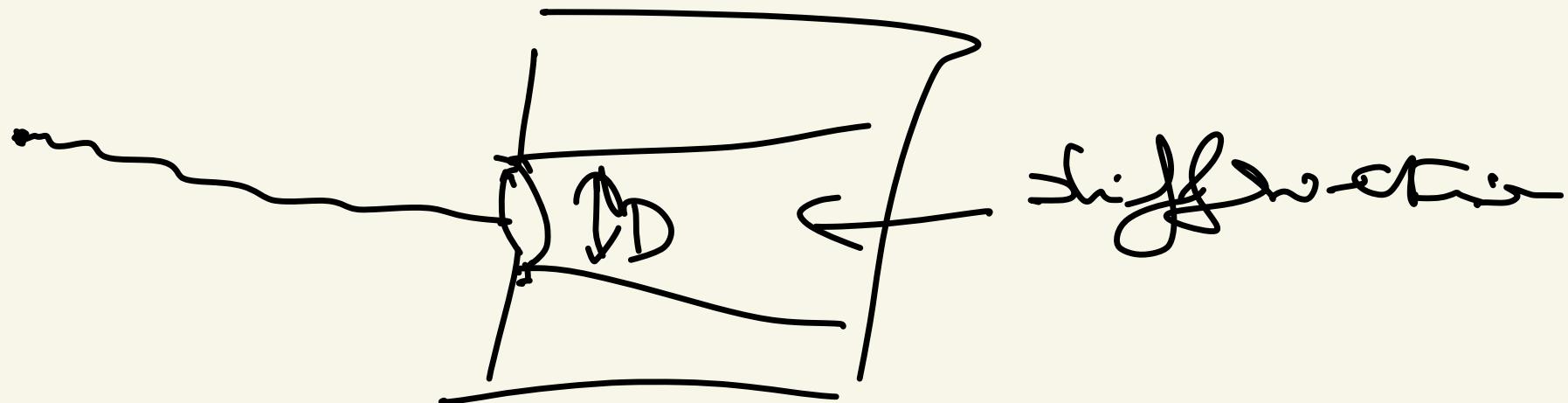
Saïd Ladjal

Interpolation

- Il s'agit de calculer la valeur de l'image en un point qui ne fait pas partie de la grille d'origine.
- On fait sur l'image, définie sur le domaine continu, une hypothèse de régularité (constante par morceaux, continue, deux fois dérivable)



\mathbb{Z}^2



$$\delta \sim \frac{\lambda}{D}$$

$$\lambda = 500 \times 10^{-9} \text{ m}$$

$$D = 2 \times 10^{-3} \text{ m}$$

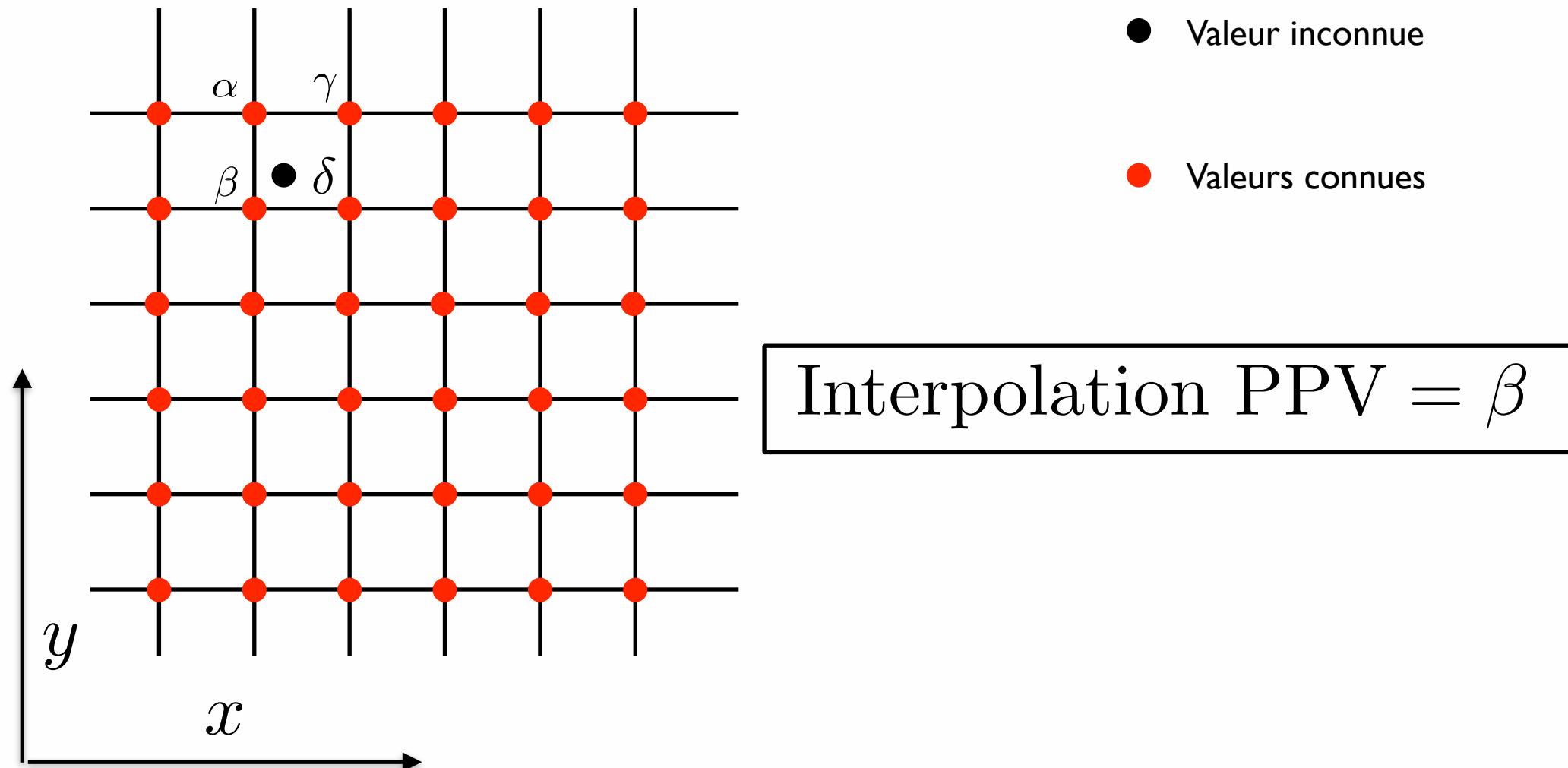
$$\delta = \frac{500 \times 10^{-9}}{2 \times 10^{-3}}$$

$$= 250 \times 10^{-6} \text{ m and } \times 4 \text{ m}$$

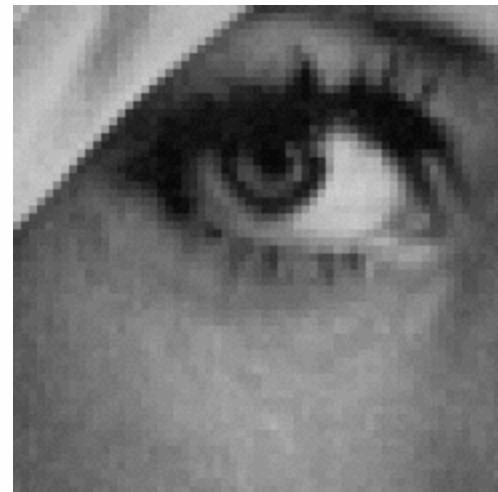
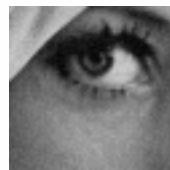
$$= 2 \times 10^{-3} \text{ m}$$

- Suivant l'hypothèse faite, on obtient des interpolations différentes.
- Constante par morceaux = plus proche voisin.
- Continue = bi-linéaire
- Polynomiale de degré 3 = bi-cubique
- À bande limitée = Interpolation de Fourier.

Exemple du plus proche voisin (PPV):



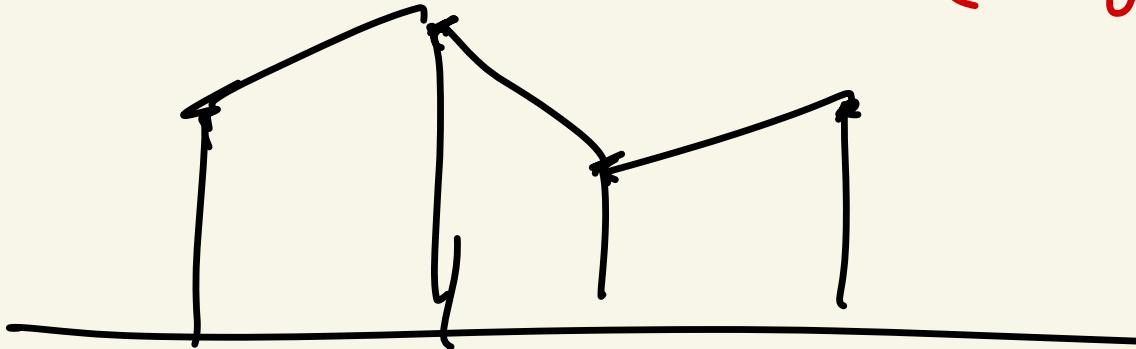
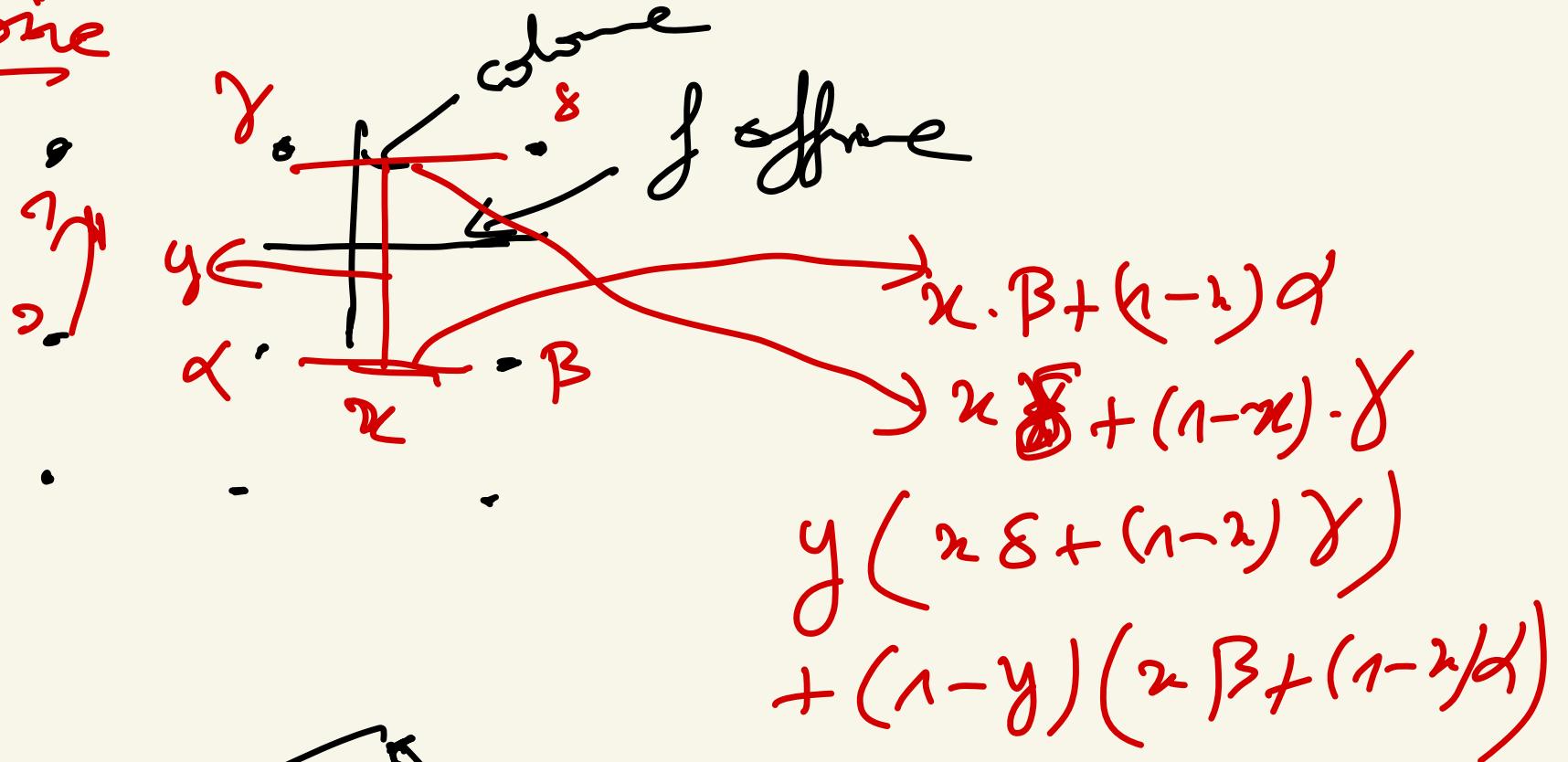
Zoom par PPV (3X)



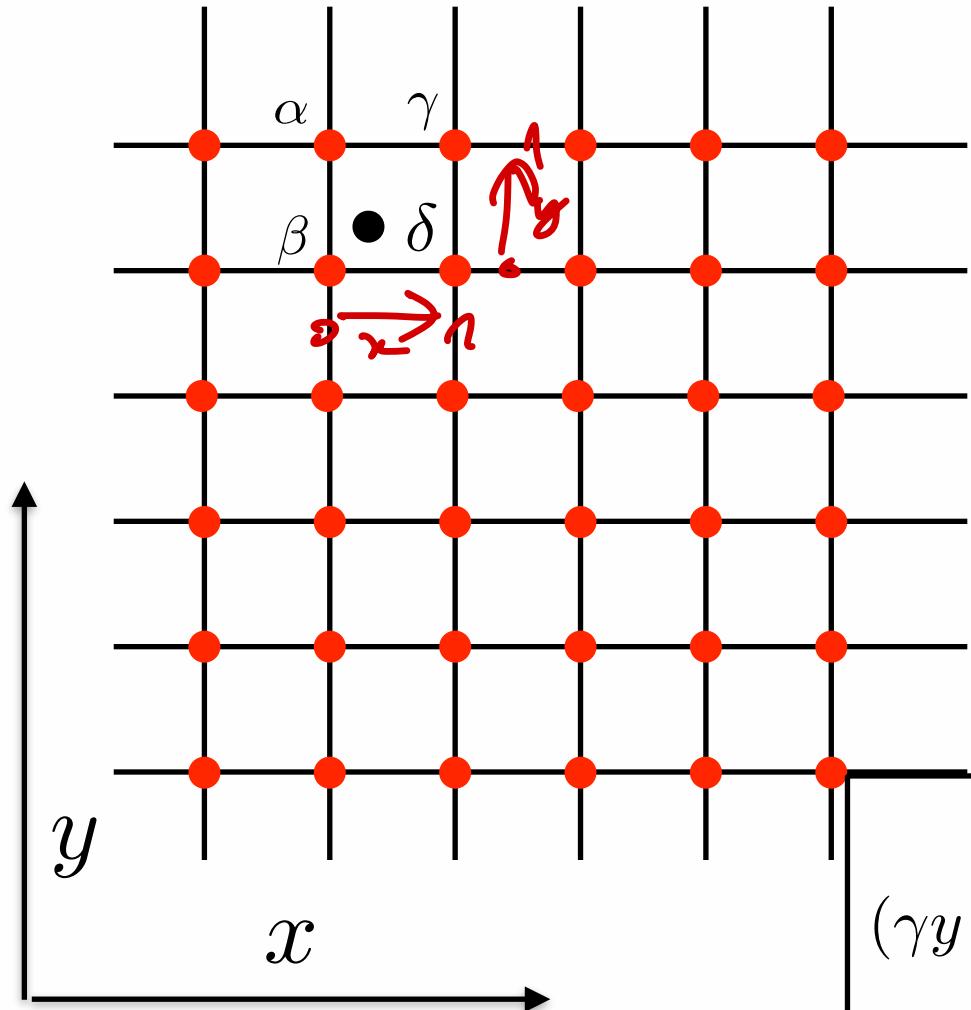
Interpolation (bi)-linéaire

- Le bi-linéaire s'obtient par interpolation linéaire dans chaque dimension successivement.

Bilinéaire



Interpolation bi-linéaire:



- Valeur inconnue

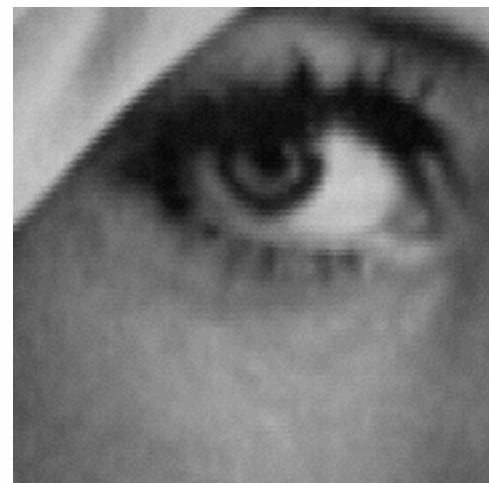
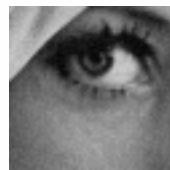
- Valeurs connues

$$\chi(y) \cdot \delta + (1-y)(1-\gamma) \beta \\ + (1-x)y \alpha + (1-x)(1-y)\gamma$$

Interpolation bi-lin =

$$(\gamma y + \delta(1 - y))x + (\alpha y + \beta(1 - y))(1 - x)$$

Zoom par bi-lin (3X)



Interpolation cubique (une dimension)

- On cherche une fonction g , polynomiale par morceaux, de degré 3 et deux fois continûment dérivable.
- Et on veut: $g(n)=f(n)$ (aux points entiers)
- Pourquoi ne pas demander 3 fois continûment dérivable?

Interpolation cubique (une dimension)

$$g(n) = f(n)$$

$g \in C^2$, g poly de deg 3

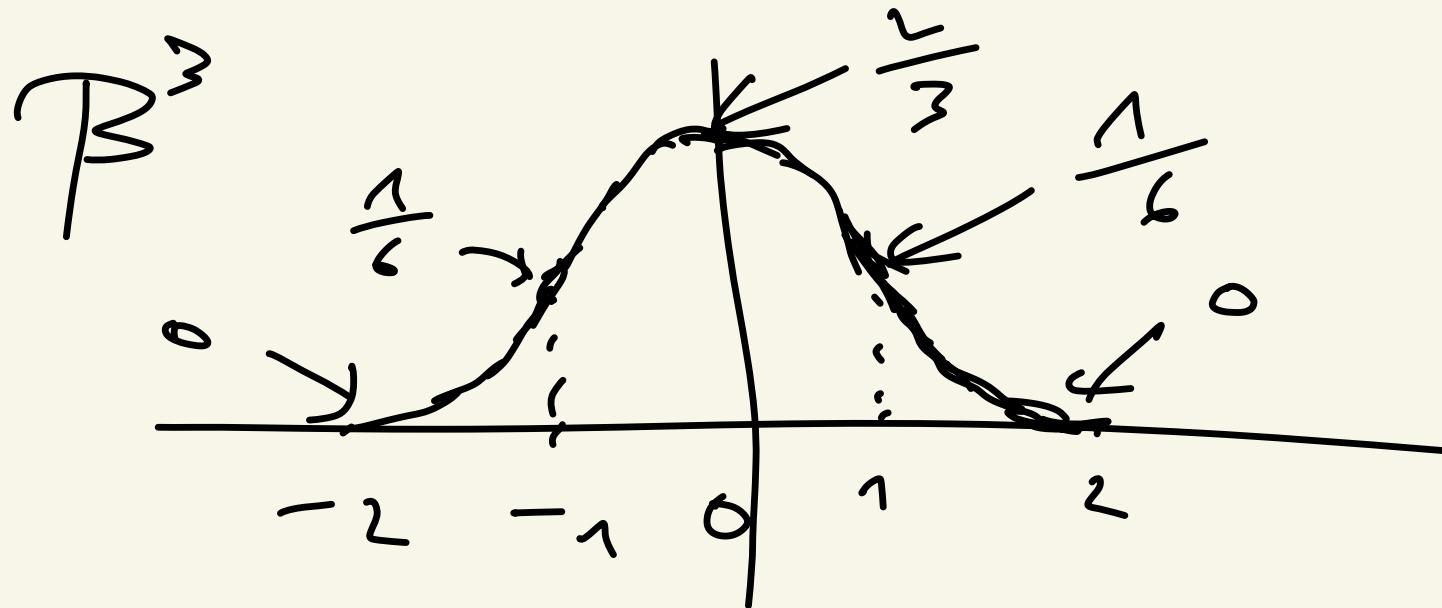
- On cherche g sous la forme:

c_k ? tq $g(n) = f(n) \rightarrow$ ~~g négé~~ ! g est négé

$$g(t) = \sum_{k \in \mathbb{Z}} c(k) \beta^3(t - k)$$

$$\beta^3(t) = \begin{cases} 2/3 - |x|^2 + |x|^3/2 & |x| \leq 1 \\ (2 - |x|)^3/6 & 1 \leq |x| \leq 2 \\ 0 & |x| > 2 \end{cases}$$

Tous les choix de $c(k)$ donnent une fonction polynomiale par morceaux et deux fois dérivable.



$$g(n) = f(n)$$

$$\left(\frac{1}{6}, \frac{2}{3}, \frac{1}{6}\right) * \left(c^n\right) = f(n)$$

$f(n) = c_{n-1} \frac{1}{6} + \frac{2}{3} c_n + \frac{2}{6} c_{n+1}$

$$\frac{1}{6}3^{-1} + \frac{2}{3} + \frac{1}{6}3 \neq 0 \quad f(3) = 1$$

$$\left[\sum c_n 3^{-n} = \frac{\sum f(n) 3^{-n}}{\left(\frac{1}{6}3^{-1} + \frac{2}{3} + \frac{1}{6}3 \right)} \right]$$

$$g(t) = \sum c_n 3^n (t-n)$$

4 operations per point
 $(c_n \text{ only})$

Interpolation cubique (une dimension)

- Comment choisir les $c(k)$? Il faut écrire les équations $f(n)=g(n)$. Cela se traduit par

$$f_n = c * b$$

$$B(z) = \frac{z^{-1} + 4 + z}{6} = \frac{1}{-6z_1} (1 - z_1 z^{-1})(1 - z_1 z)$$

$$z_1 = -2 + \sqrt{3} \quad |z_1| < 1$$

$$b = \left(\frac{1}{c}, \frac{2}{3}, \frac{7}{6} \right)$$

$$\frac{1}{(1 - z_1 z^{-1})}$$

Interpolation cubique (une dimension)

$$y_n + 3z_1 y_{n-1} = x_n$$

$$y_n = x_n - z_1 y_{n-1}$$

- On applique deux filtres récursifs: l'un causal et l'autre non causal pour obtenir les $c(k)$ au prix de quelques opérations par échantillon.

Signal de taille N

4 N opération $\rightarrow c_k$

4 x nb d'interpolation qui les c_k sont
couvrus.

Interpolation cubique (deux dimensions)

- Dans le cas bi-dimensionnel on cherche g sous la forme

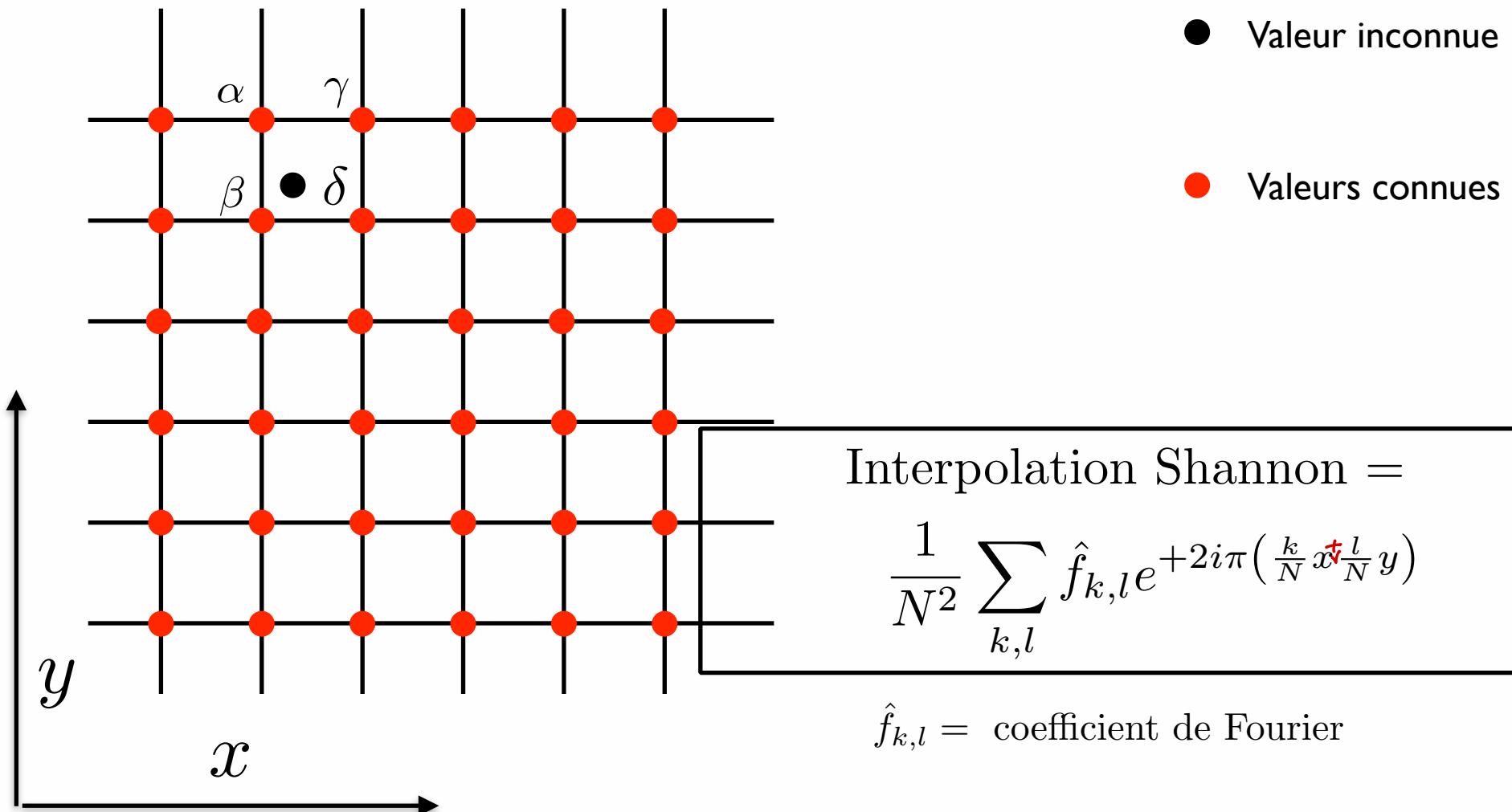
$$g(x, y) = \sum_{k,l} c(k, l) \beta^3(x - k) \beta^3(y - l)$$

Les $c(k, l)$ sont obtenus de manière séparable: On les calcule pour chaque ligne suivant le procédé mono-dim. Puis on applique le processus mono-dim le long des colonnes.

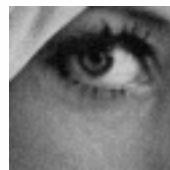
Zoom par bi-cubique (3X)



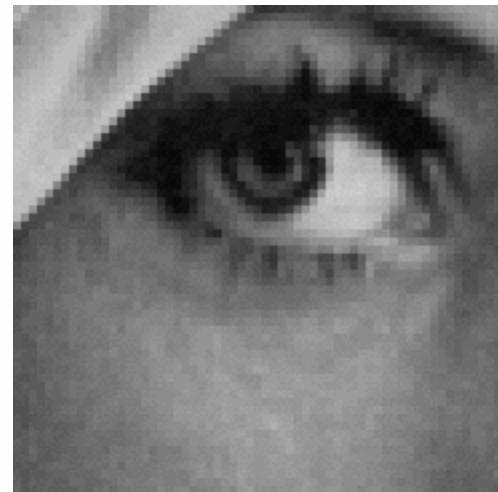
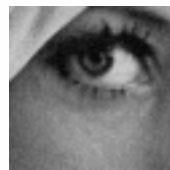
Interpolation Shannon:



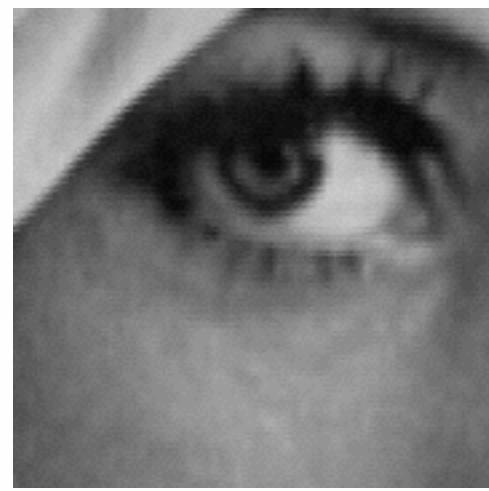
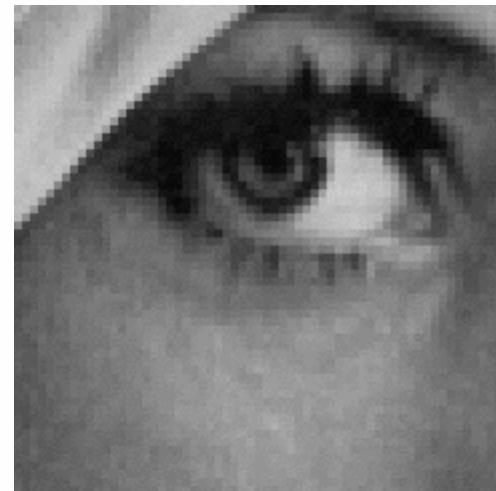
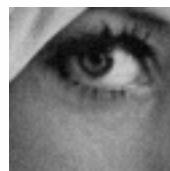
Zoom par Shannon (3X)



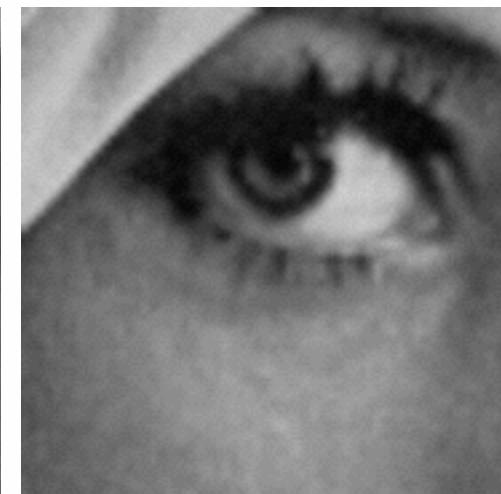
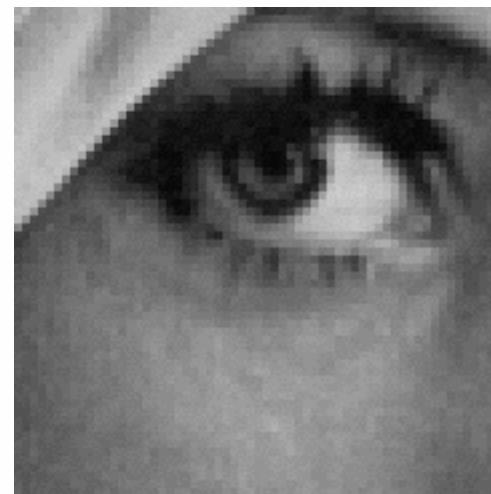
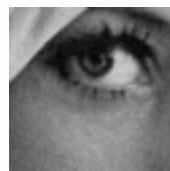
Zoom par PPV (3X)



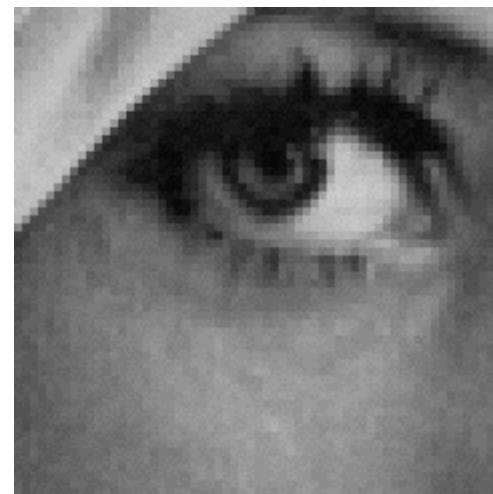
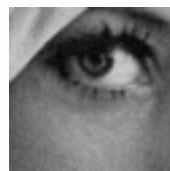
Zoom par bi-lin (3X)



Zoom par bi-cubique (3X)

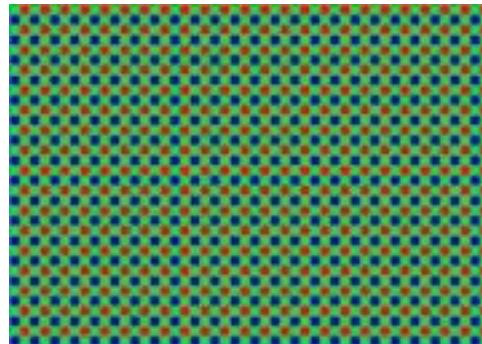


Zoom par Shannon (3X)

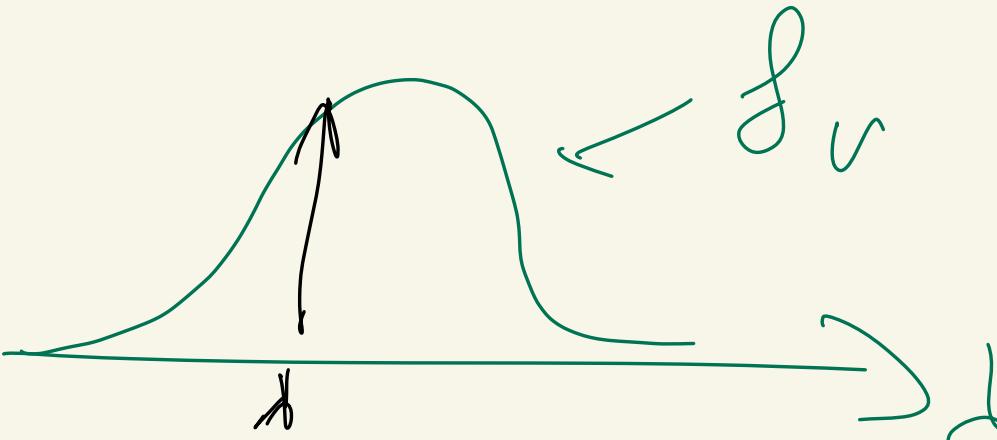


Interpolation pour le démosaïcage

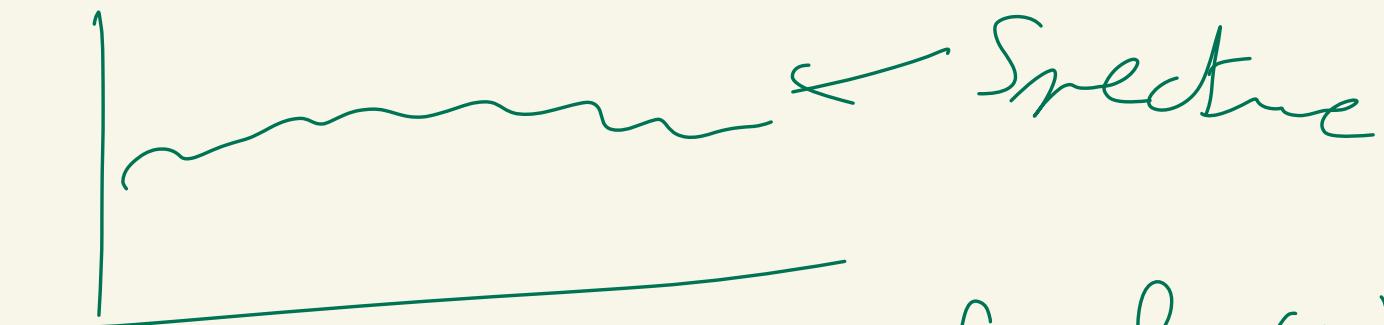
- Contrairement au scanner, l'appareil photo doit prendre une photographie instantanée. Il ne peut donc pas photographier séquentiellement le rouge, vert et bleu.
- On a recours à une matrice CCD de Bayer



Nécessité d'un algorithme pour calculer les composantes RVB en tout point (démosaïcage).



$\langle f_v \rangle$ Speckle



$$\int f_v(x) \cdot S_p(x) dx$$

NE Visible

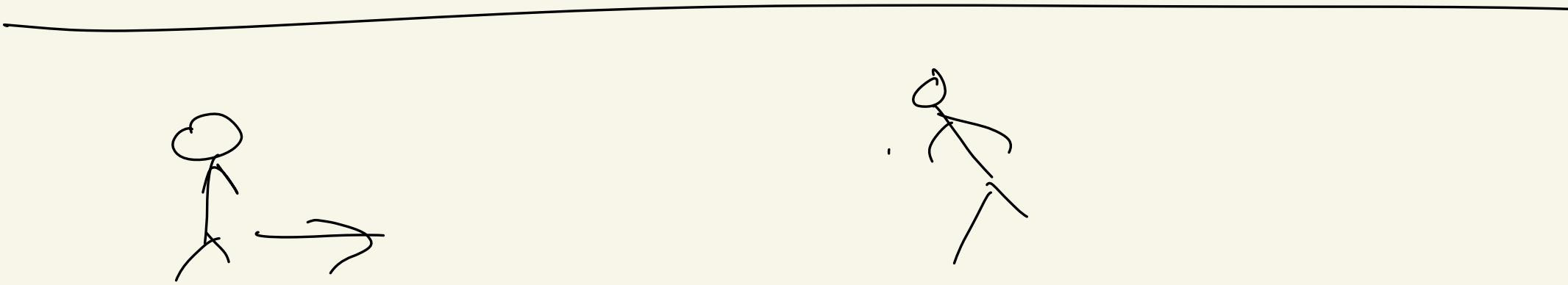
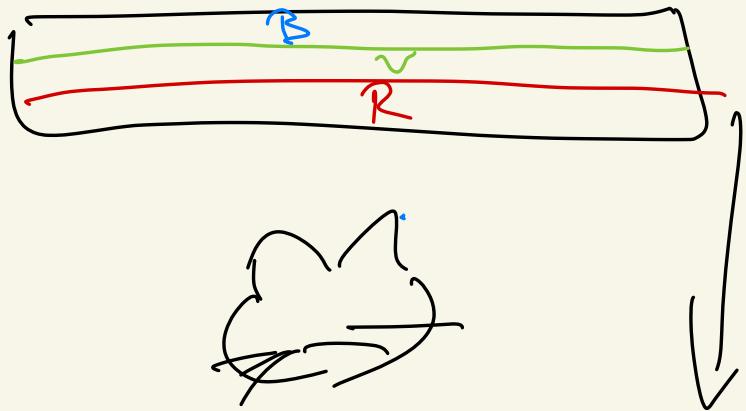
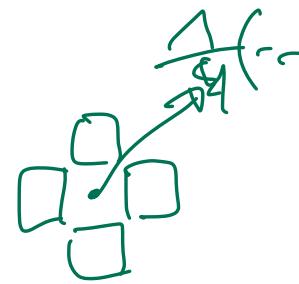
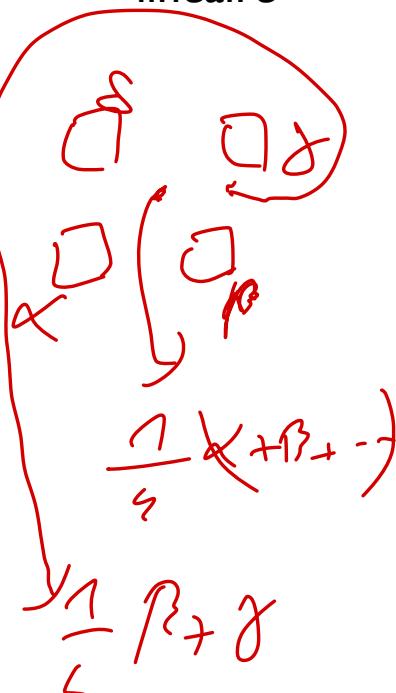


Image parfaite

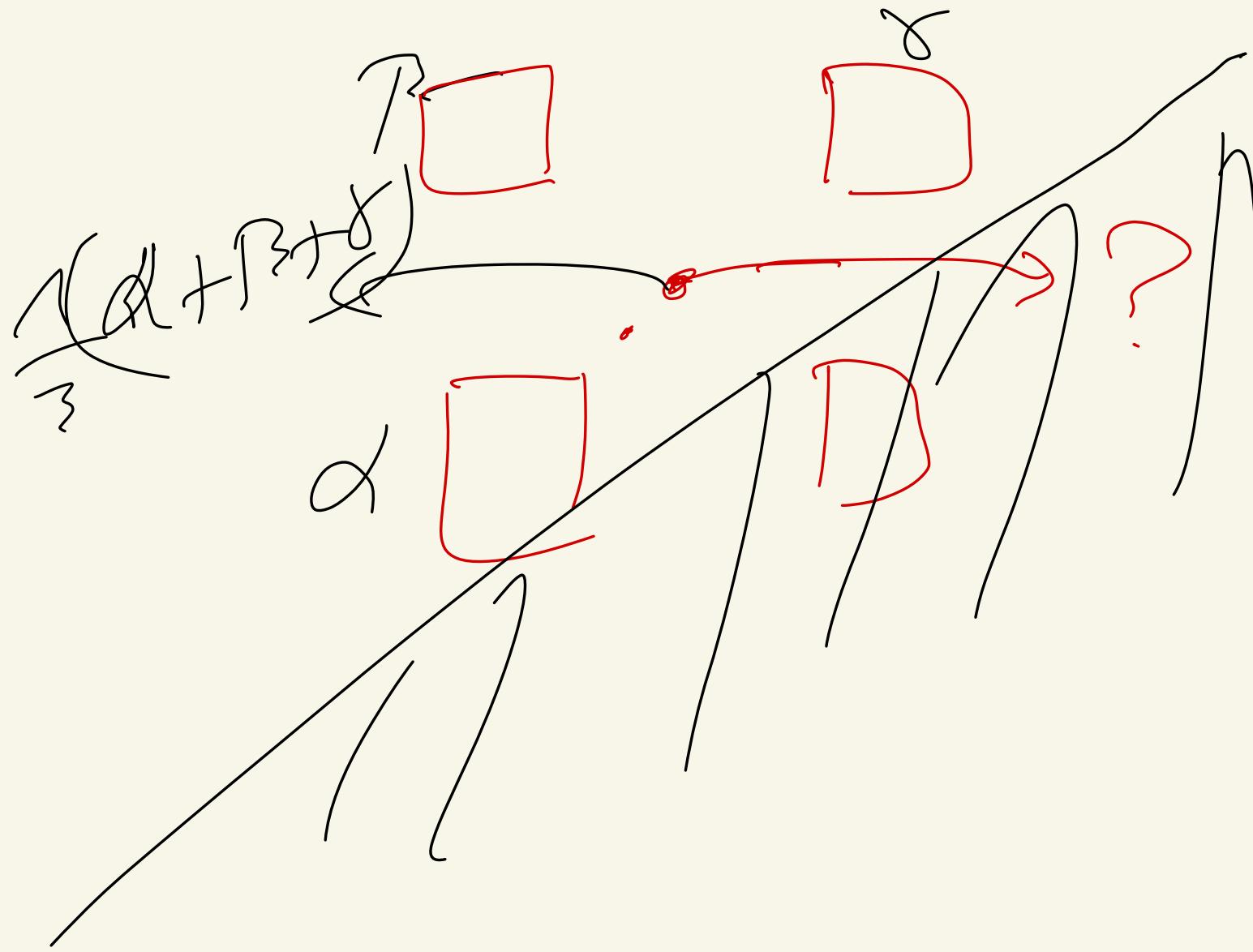


Bayer suivi d'une
interpolation bi-
linéaire

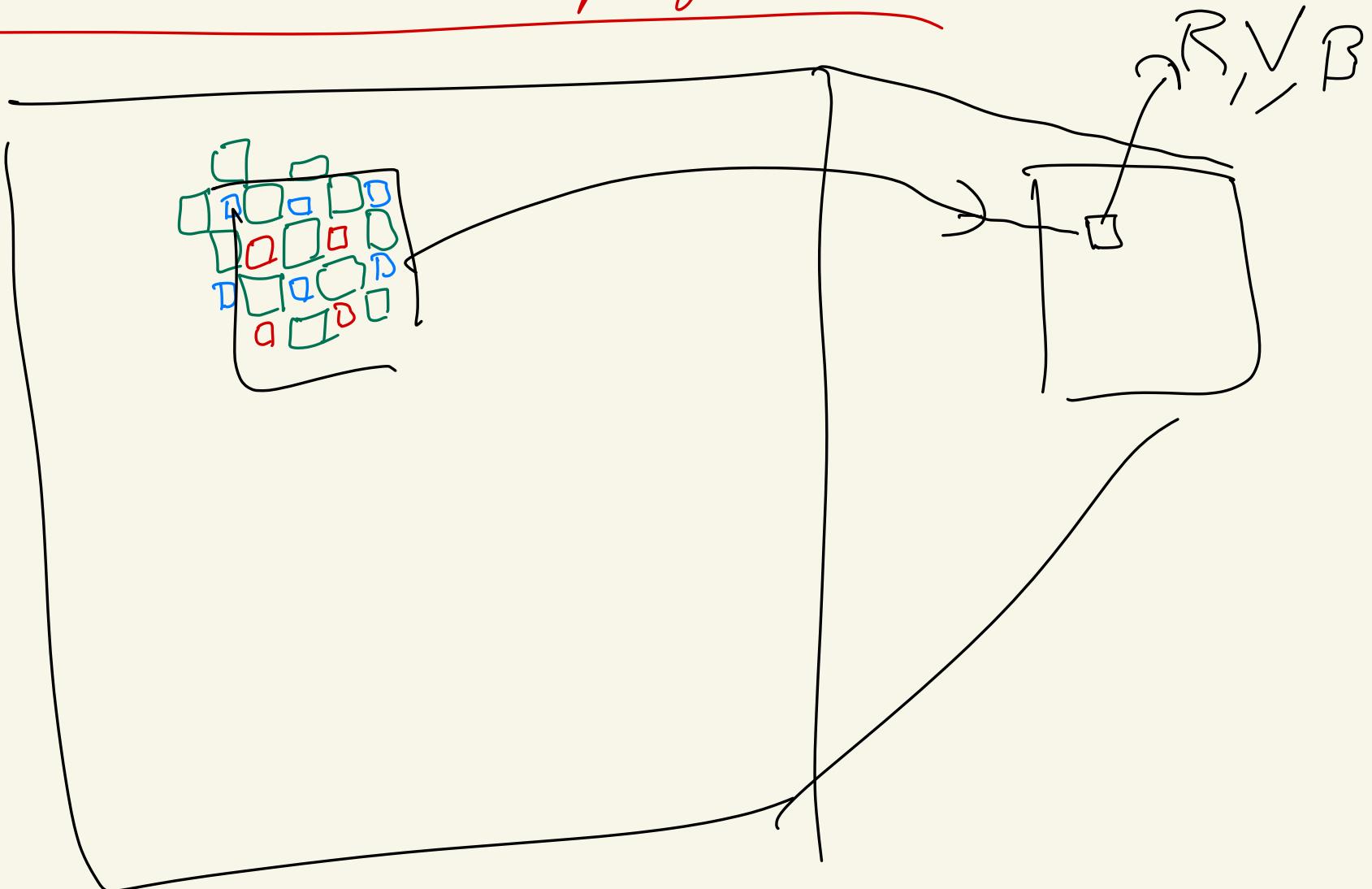


Demosaicage:



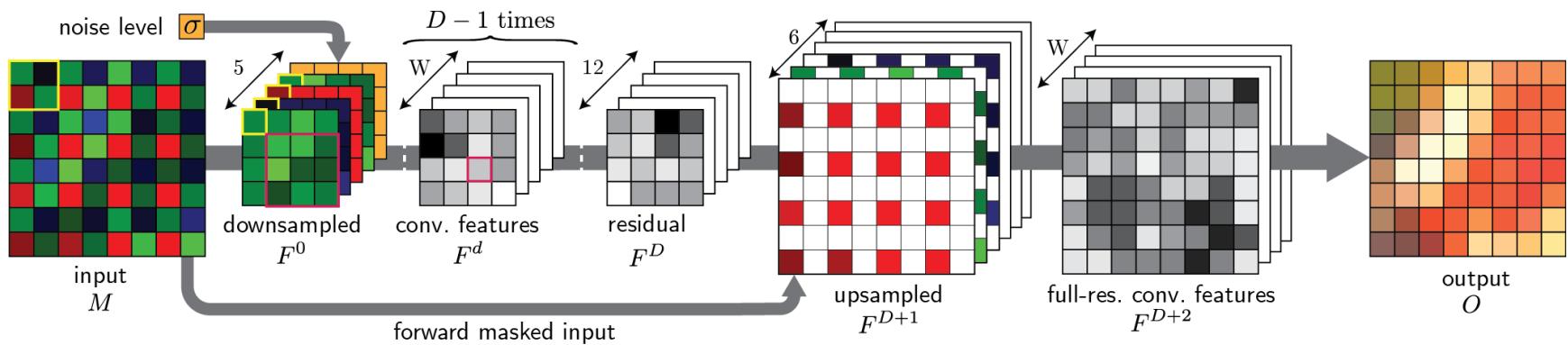


Obtener una imagen perfecta?



Interpolation pour le démosaïcage

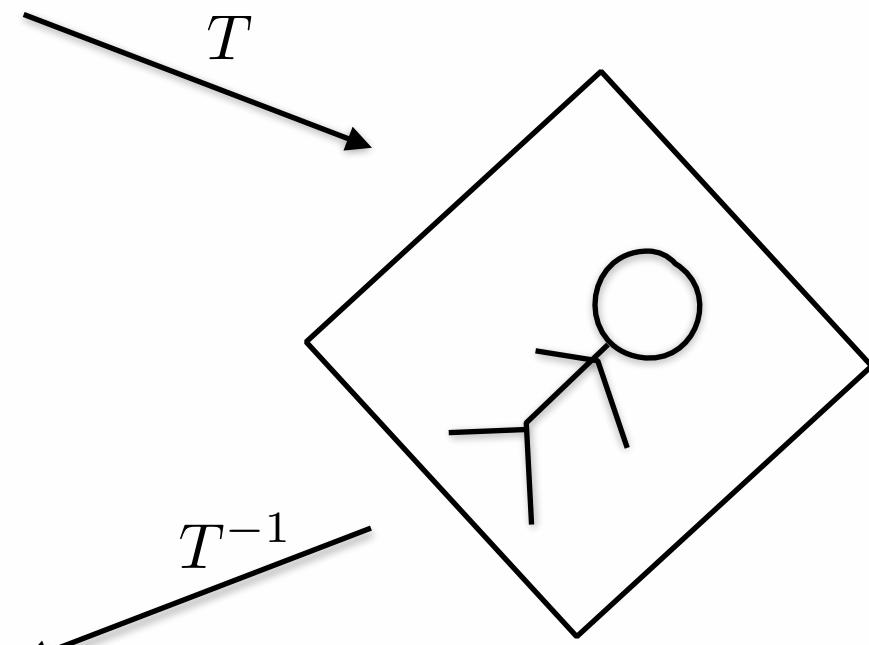
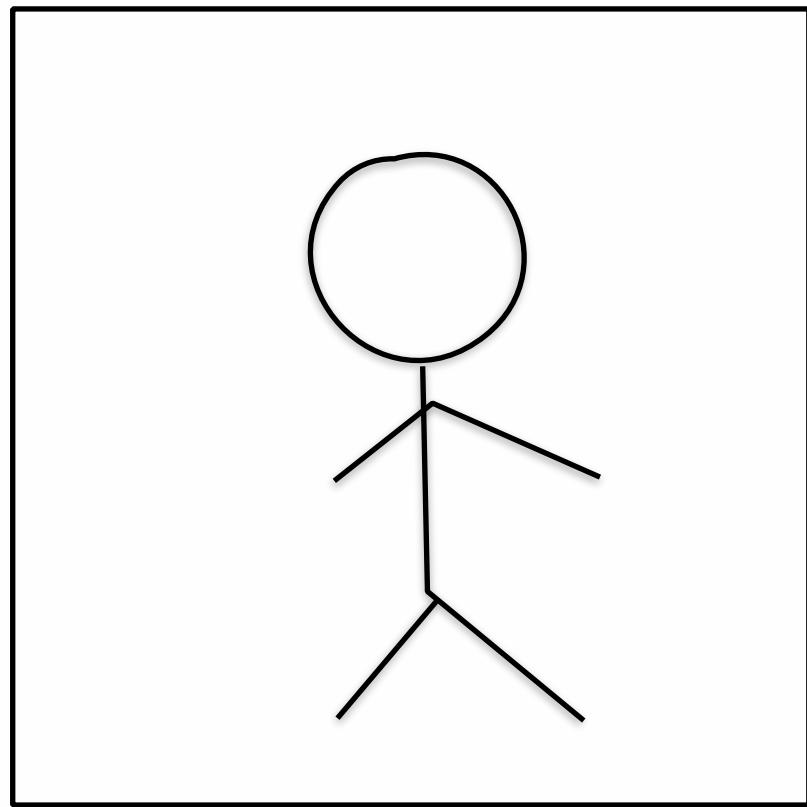
- Récemment une proposition d'architecture CNN a obtenu de très bon résultats



Le temps de calcul est trop long pour un appareil portatif

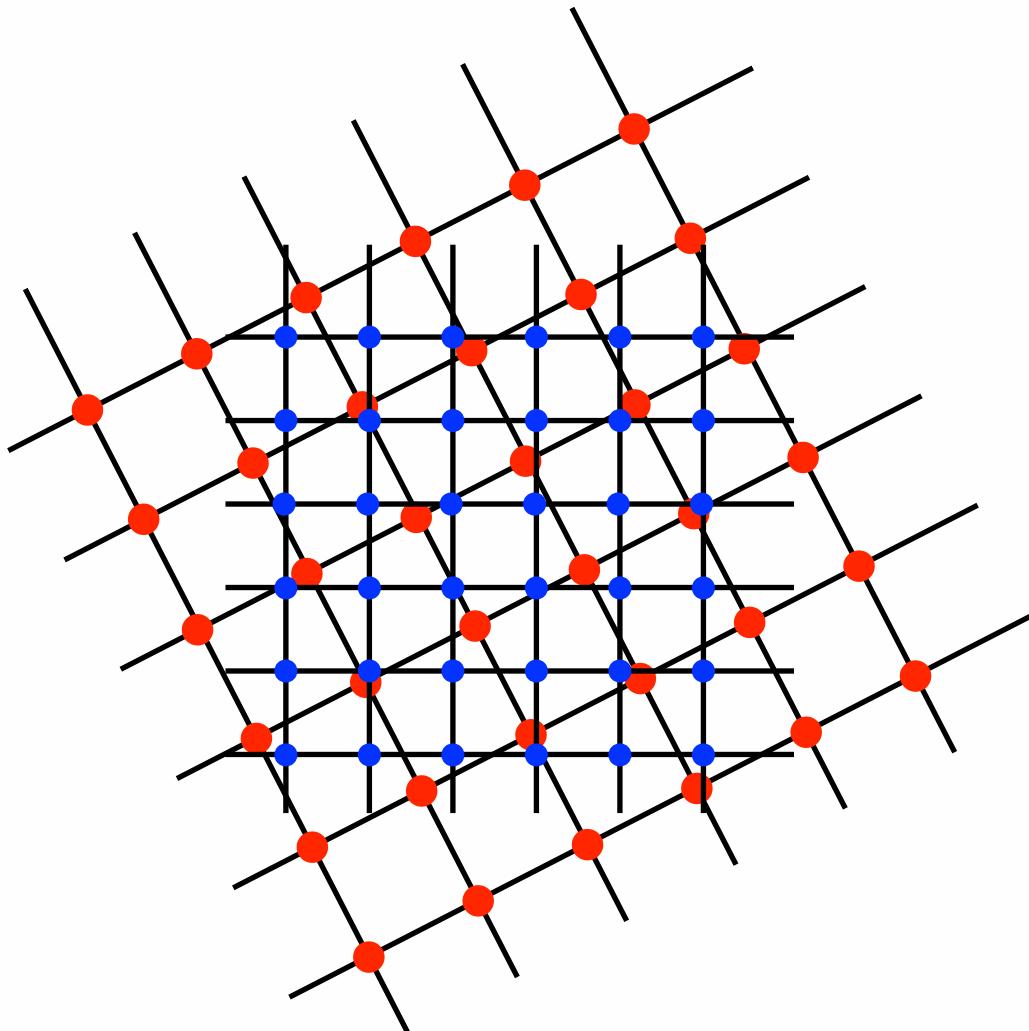
Transformations géométriques

- Pour différentes raisons, on veut faire subir à l'image une transformation géométrique.
- Pour obtenir la valeur de la nouvelle image en un pixel (i,j) , on applique l'inverse de la transformation géométrique pour trouver la position dans l'image d'origine.
- Les coordonnées trouvées ne sont pas entières d'où l'importance de l'interpolation.



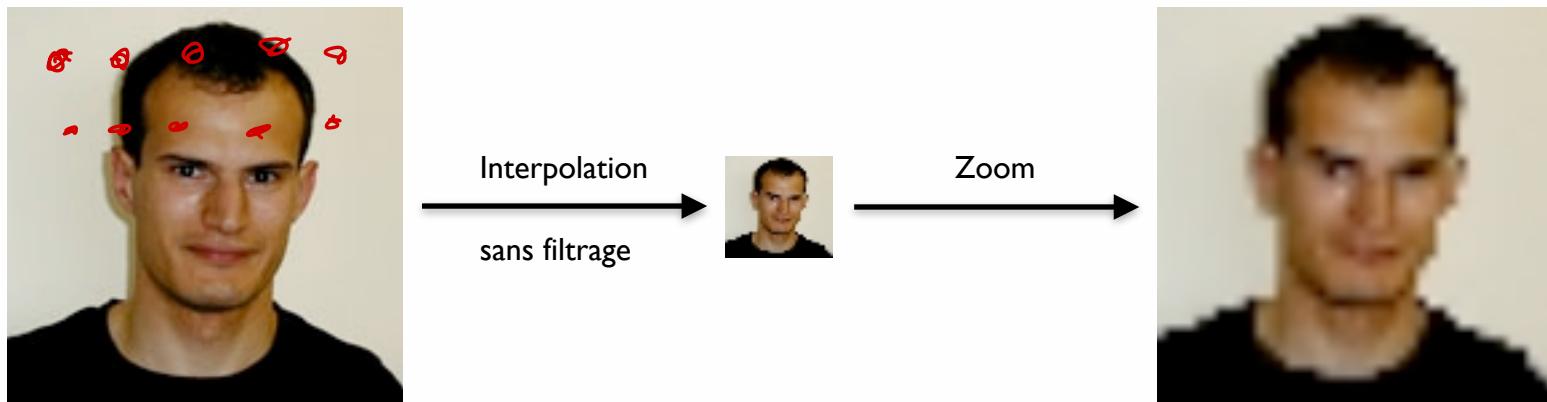
- Inconnue
- Connue

Dans le repère de l'image rouge (connue), les points entiers de l'image bleue (inconnue), ne sont pas entiers.

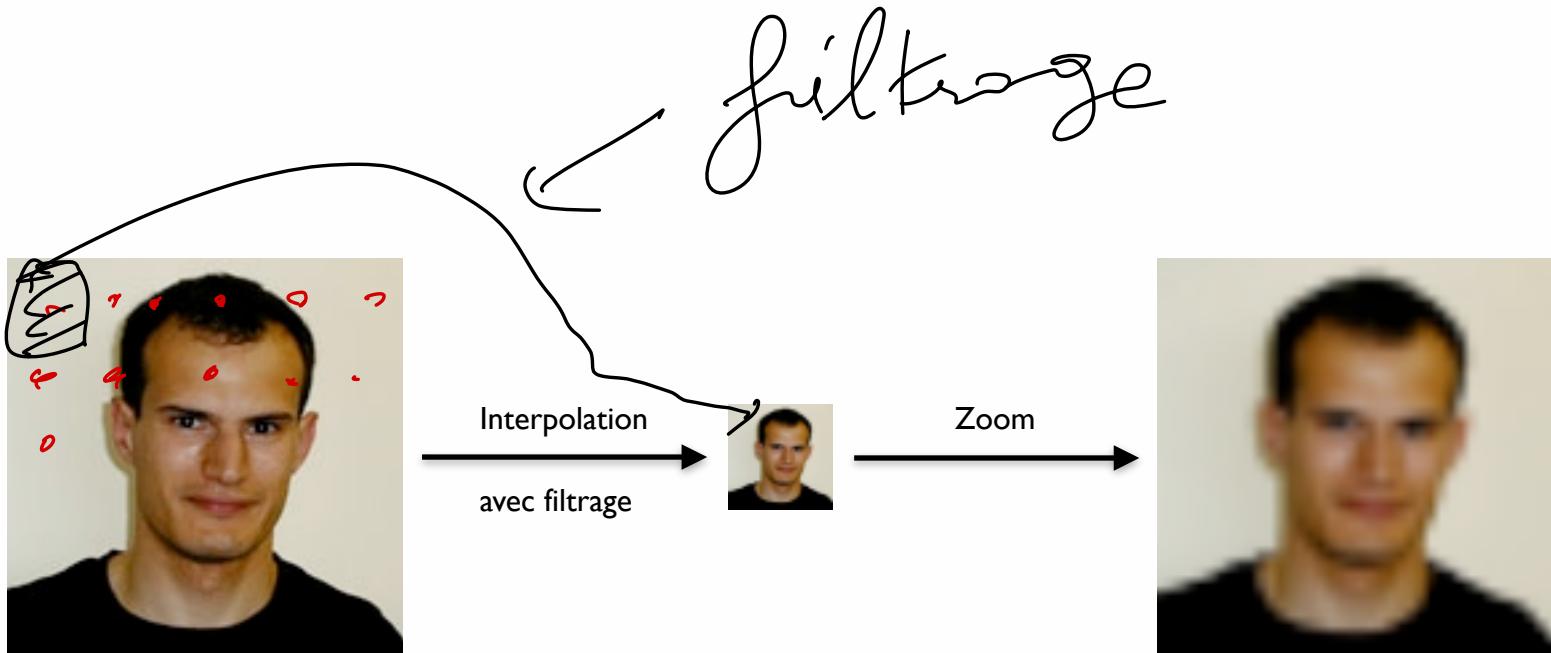


Est-ce un zoom ou un dézoom?

Cas d'une diminution de taille: Il faut appliquer un filtre passe-bas!



Cas d'une diminution de taille: Il faut appliquer un filtre passe-bas!

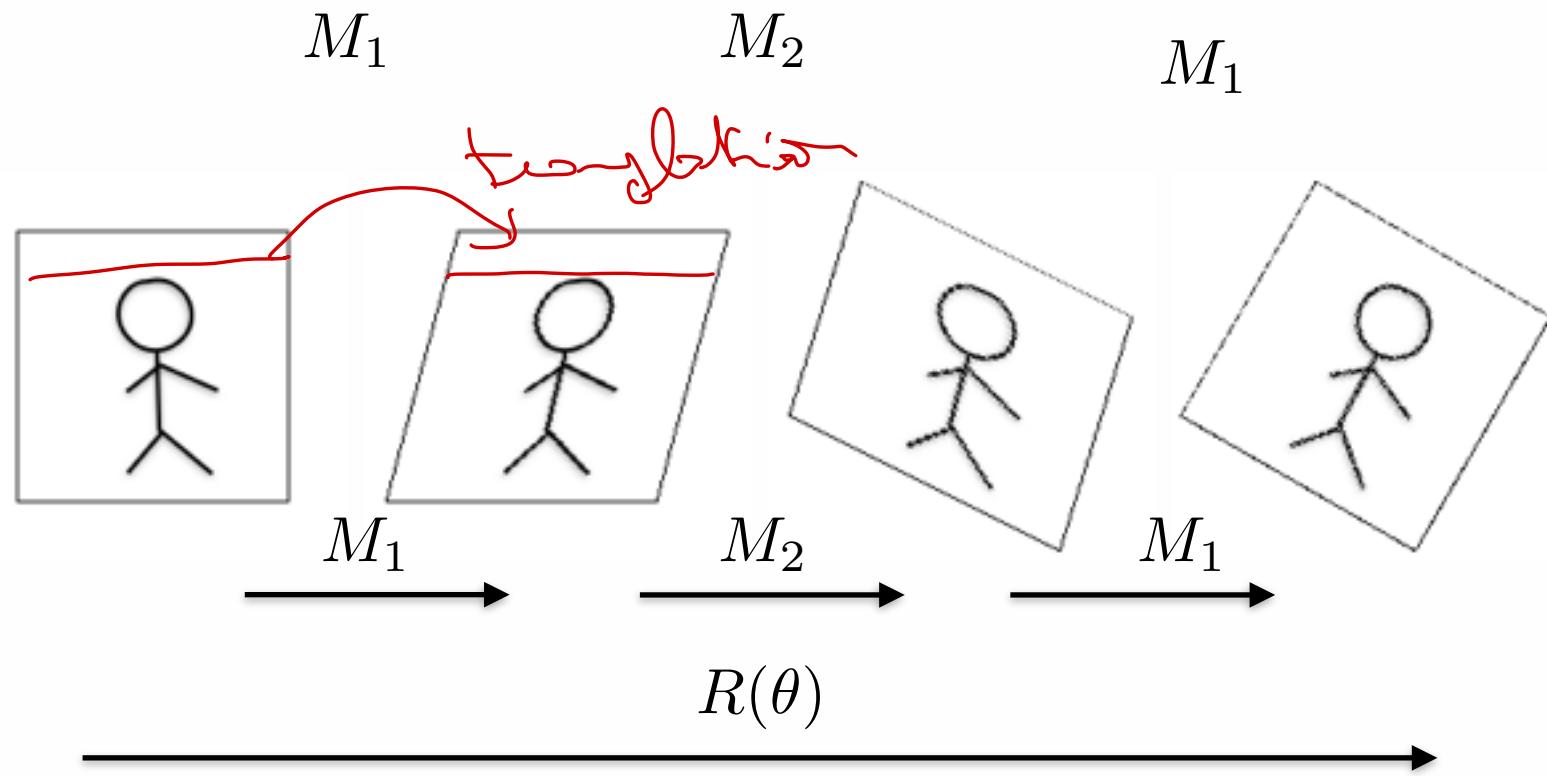




Exemple de répétition d'une transformation géométrique avec différentes interpolations.

Principe de la rotation précédente:

$$R(\theta) = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} =$$
$$\begin{bmatrix} 1 & -\tan \frac{\theta}{2} \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 \\ \sin \theta & 1 \end{bmatrix} \times \begin{bmatrix} 1 & -\tan \frac{\theta}{2} \\ 0 & 1 \end{bmatrix}$$



$$f(t) = \sum \hat{f}_k \cdot e^{2\pi i n t k}$$

$$\begin{aligned} f(t+s) &= \sum_k \hat{f}_k e^{-i\pi(k+s)k} \\ &= \sum \left(\hat{f}_k \cdot e^{-i\pi s k} \right) \cdot e^{2\pi i t k} \\ &= \overline{\sum_k \hat{f}_k} \cdot e^{-i\pi s k} \end{aligned}$$

$$s = \alpha y$$

Le filtrage

- Le filtrage est une opération qui transforme une image en une autre image de même taille.
- Pour chaque pixel de la nouvelle image on décide d'une valeur à partir des anciennes valeurs de l'image.

Filtrage linéaire

- Filtrage linéaire (invariant par translation): Il revient à une convolution par un masque.
- Suivant la taille du masque et de ses valeurs on obtient des filtrages différents.
- Filtre moyenne=masque constant.
- Filtre gaussien=masque gaussien.
- On distingue souvent les filtres passe-haut et passe-bas

$$f * \begin{bmatrix} \frac{1}{N^2} & \cdots & \frac{1}{N^2} \\ \cdots & \cdots & \cdots \\ \frac{1}{N^2} & \cdots & \frac{1}{N^2} \end{bmatrix}$$

N est la taille du filtre



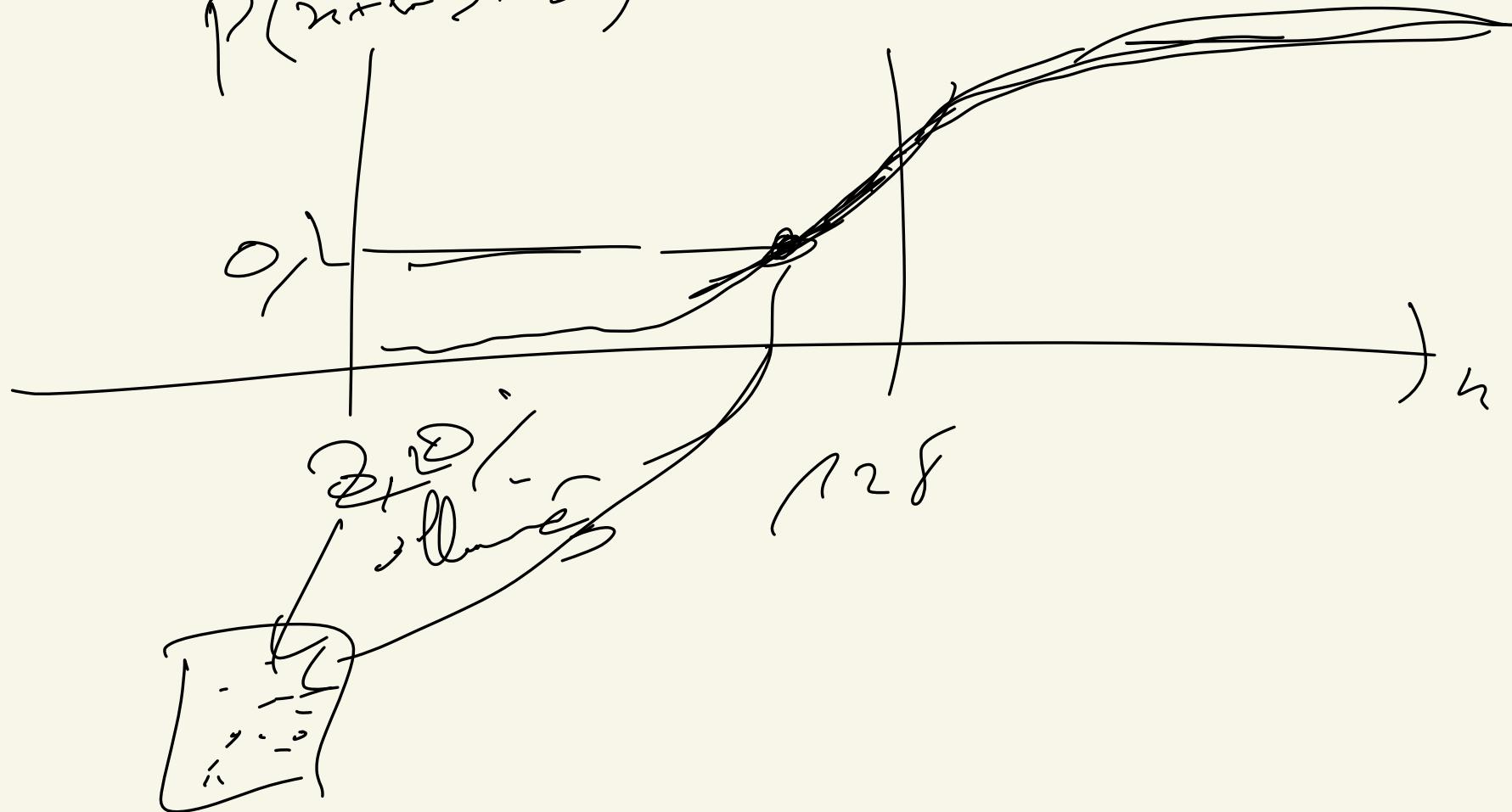
Exemple de filtrage moyenne (avec accroissement de la taille du masque)

$$x + b > 128 \rightarrow 1$$

DITHERING

$$x + b \leq 128 \rightarrow 0$$

$$P(x+b > 128)$$



Filtrage linéaire: Passe haut: exemple du gradient

$$g_x(i, j) = f(i, j) - f(i - 1, j) = f * [1, -1]$$

$$g_y(i, j) = f(i, j) - f(i, j - 1) = f * \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

$| \partial_x |$ ↗



g_x



$| \partial_y |$ ↗



g_y

Les filtres pour le débruitage:

I) Filtre moyenne



Un effet de flou est introduit. Il est dû au fait que différentes valeurs sont moyennées au travers des transitions de l'image.
Plus généralement, on a moyenné des valeurs qui ne sont pas de même nature que le pixel central.

Les filtres pour le débruitage: II) Filtre médian

Remplacer la valeur moyenne dans un voisinage par la valeur médiane.
Il est particulièrement efficace pour le bruit impulsif.



10% des pixels perdus



Médian
3x3



Moyenne
3x3



Médian
5x5



Moyenne
5x5

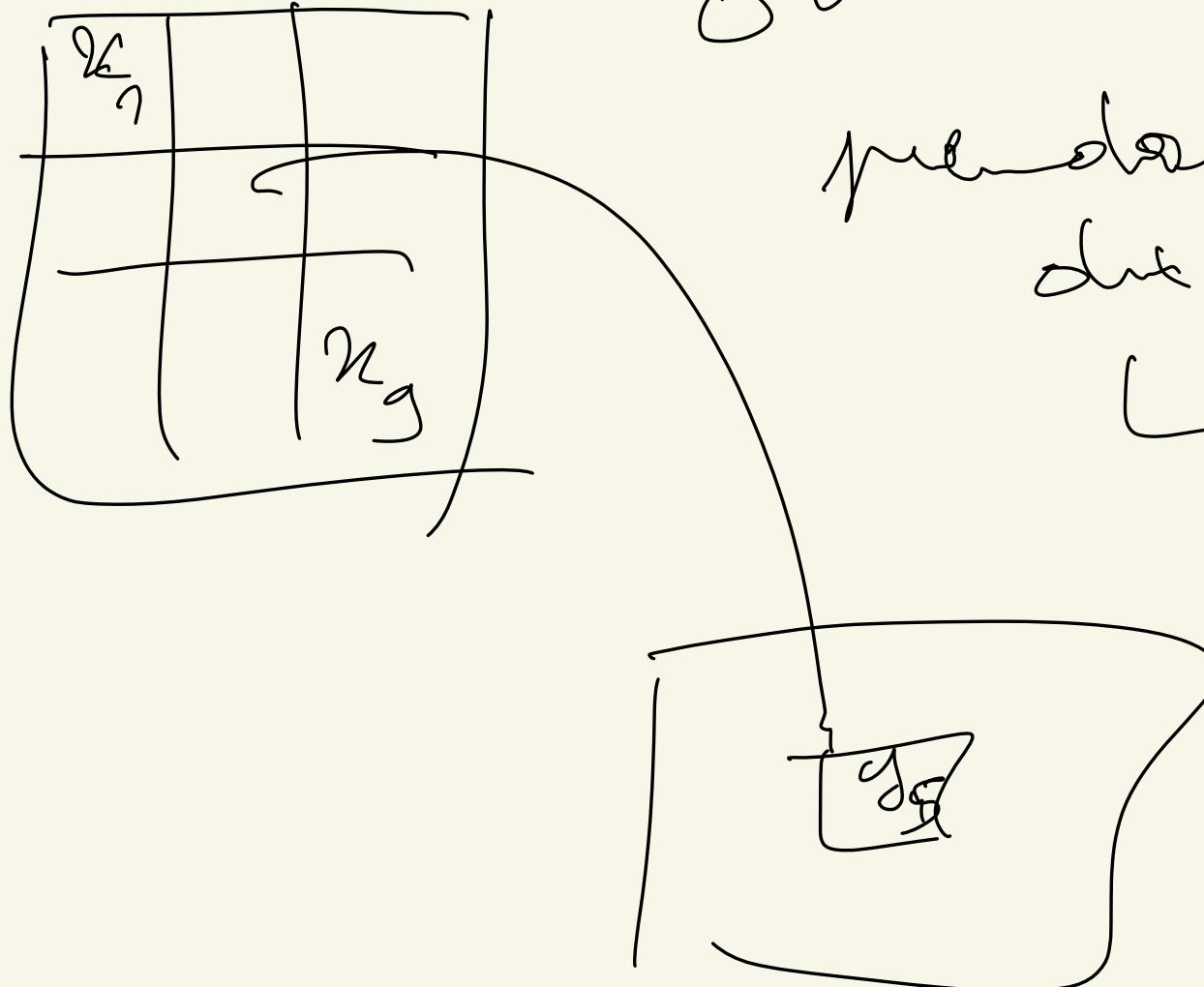
Filtre médian

$$y_1 \leq y_i - \epsilon y_g$$

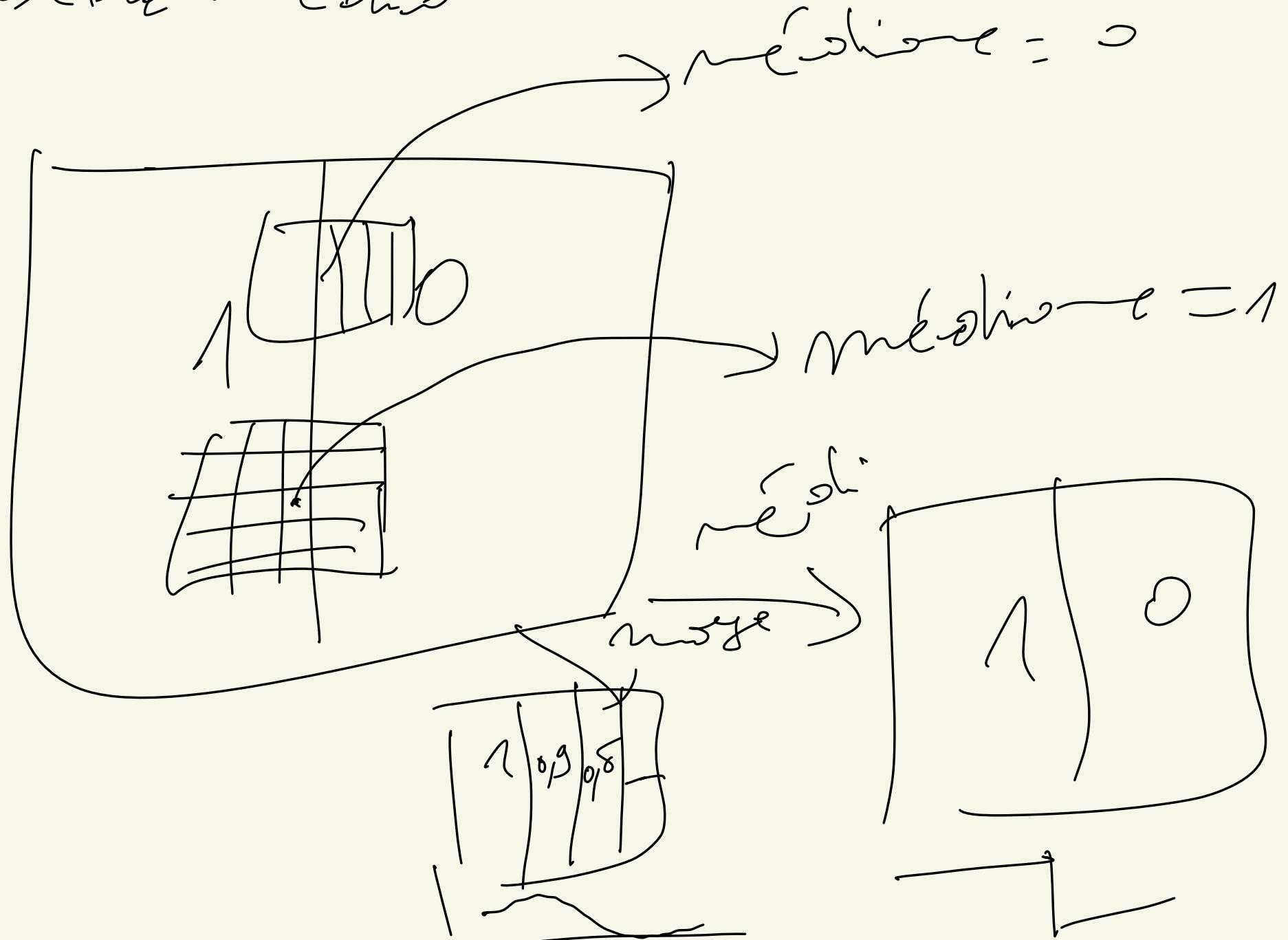
croissance y_1, \dots, y_g

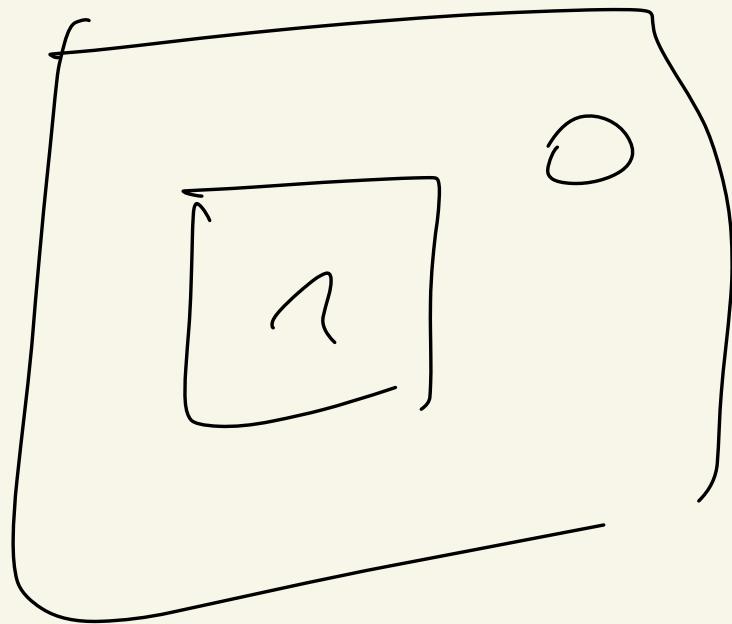
prendre le volume
des particules

$$\sum y_i$$

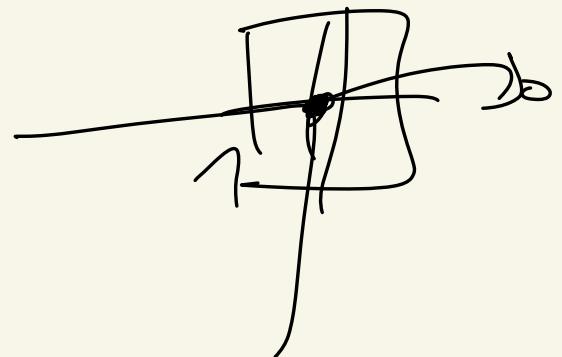
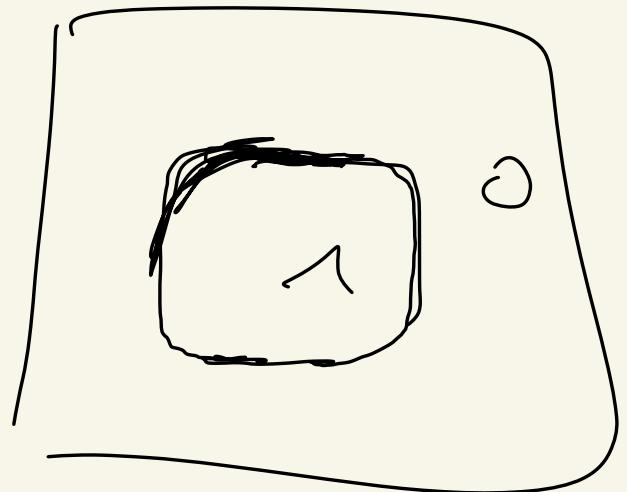


Filtre médian:





→ *mislied*



Les filtres pour le débruitage: III) Filtre bilateral

$$g(i, j) = \frac{1}{K} \sum_{k,l} f(k, l)w(k, l)$$

$$w(k, l) = S(k - i, l - j)I(f(k, l) - f(i, j))$$

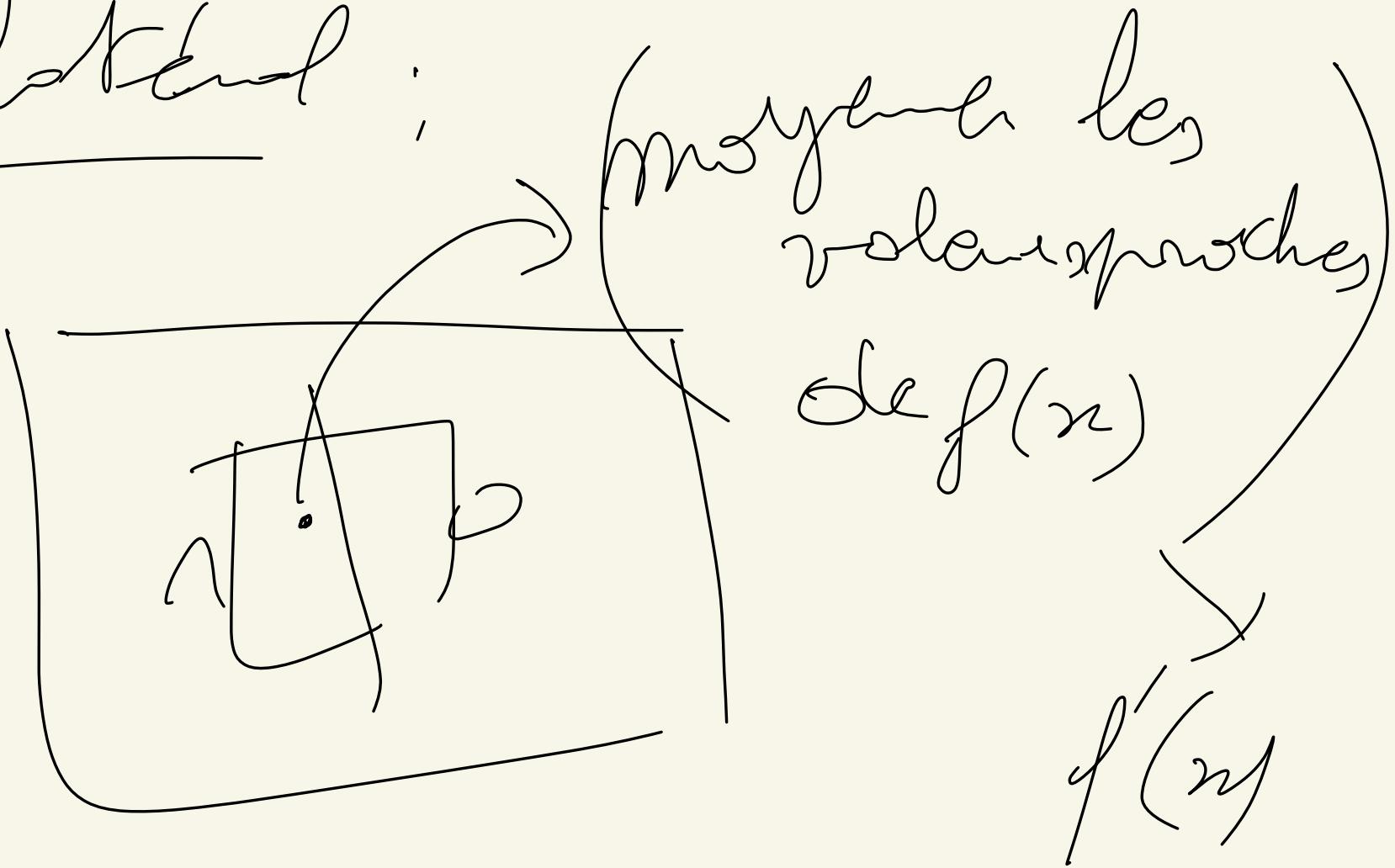
$$I(t) = e^{-\frac{t^2}{2\sigma_I^2}}$$

$$S(x, y) = e^{-\frac{x^2+y^2}{2\sigma_S^2}}$$

$$K = \sum_{k,l} w(k, l)$$

Idée: Prendre plus en compte dans la moyenne les pixels qui sont proches spatialement et en niveau de gris.

Bildschau:



Les filtres pour le débruitage: III) Filtre bilateral



Respecte bien les bords, mais floute encore les zones de texture.

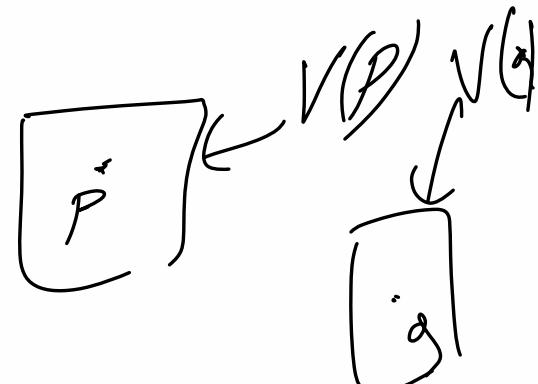
Les filtres pour le débruitage: IV) Moyennes non locales

- Au lieu de faire une moyenne locale de l'image afin d'atténuer le bruit, on moyenne les valeurs de pixels dont l'entourage ressemble à celui du pixel traité:

$$nouveau(p) = \sum_q ancien(q) \frac{D(V(p), V(q))}{\sum_q D(V(q), V(p))}$$

$V(n)$ voisinage du pixel n

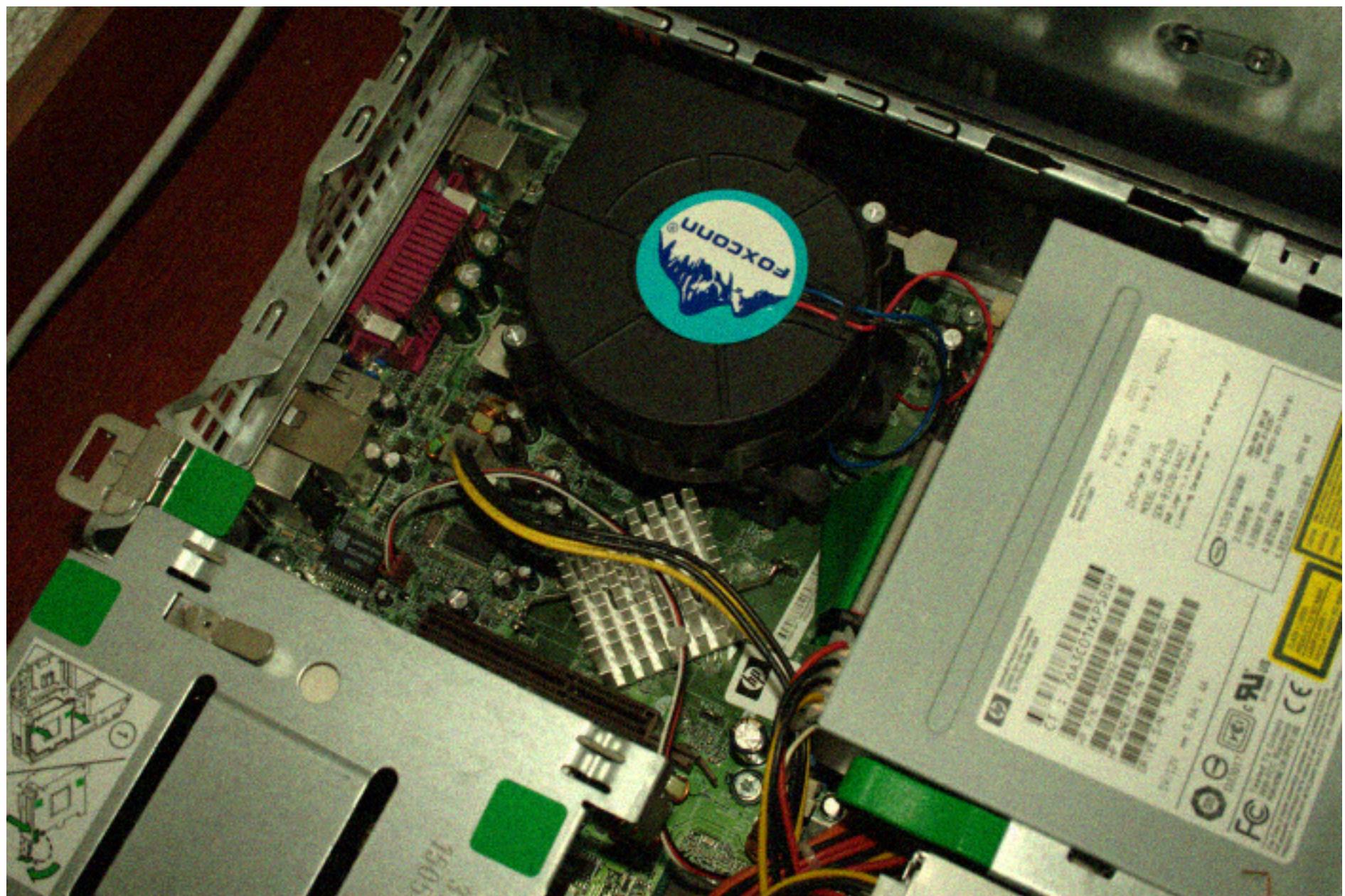
$$D(V1, V2) = e^{-\frac{\|V1 - V2\|_2^2}{h^2}}$$

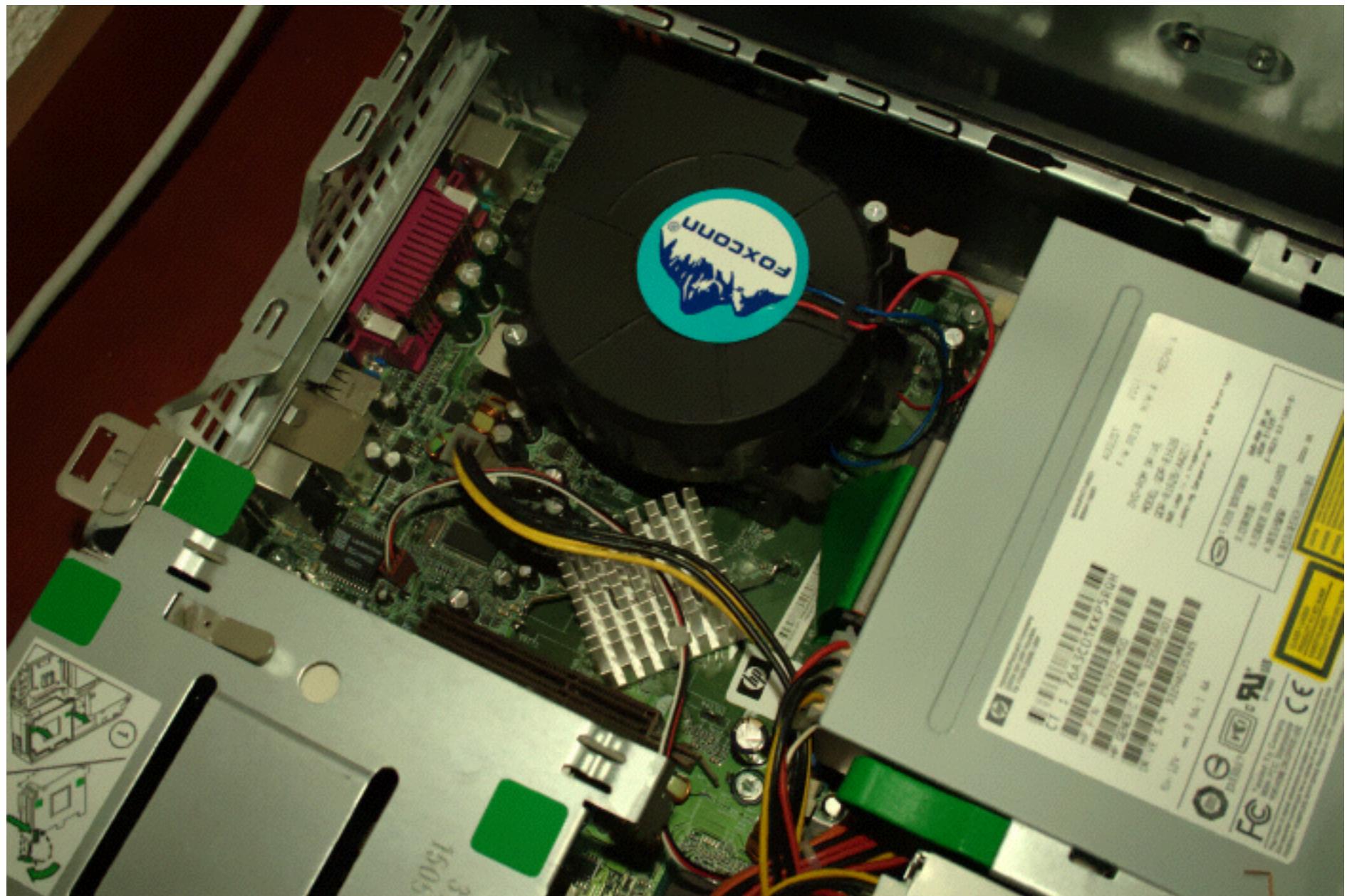


Les filtres pour le débruitage: IV) Moyennes non locales









Restoration: les images à deux pixels.

$$\begin{pmatrix} x, y \end{pmatrix} \xrightarrow{\text{photo}}$$

$$x = 3, y = 2$$

$$\begin{pmatrix} 1 & 0 \\ 0 & \omega^{-6} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} 0, 01 \\ 0, 02 \end{pmatrix}$$

↑
 M

bruit

obsr: $x' = 3, 01$

$$y' = 0, 02 + 2 \times \omega^{-6}$$

bruit

$$M^{-1} \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} 3, 01 \\ 2 + 2 \times \omega^4 \end{pmatrix}$$

\rightarrow bruit

$$\begin{pmatrix} u' \\ y' \end{pmatrix} = M \begin{pmatrix} u \\ y \end{pmatrix} + b$$

$$\text{et } \|b\| \leq \epsilon^2$$

$$\left\| M \begin{pmatrix} u \\ y \end{pmatrix} - \begin{pmatrix} u' \\ y' \end{pmatrix} \right\| \leq \epsilon^2$$

