Outils et traitements de bases

– Partie 2 –

Bibliographie

Ouvrages :

→ *Digital Image Processing, 3rd Ed.*, Rafael C. Gonzalez and Richard E. Woods, Prentice Hall, 2008.

Cours :

→ Vincent Mazet, cours "Outils fondamentaux pour le traitement d'image", *http ://miv.u-strasbg.fr/mazet/ofti*

→ Vincent Noblet, cours "Traitement d'images" TICS2A, *http ://icube-miv.unistra.fr/fr/index.php/Traitement_d'images_TICS2A*

Plan du chapitre

Convolution

Beaucoup de traitements s'obtiennent en modifiant les valeurs des pixels en fonction de leurs pixels voisins.

Lorsque cette modification est identique à toute l'image, elle peut être définie à l'aide d'une seconde image qui définit les relations de voisinage.

Il en résulte une nouvelle image J calculée à partir de l'image originale I et du voisinage H :

$$J(x, y) = (I * H)(x, y) = \sum_i \sum_j H(i, j) I(x - i, y - j)$$

Convolution

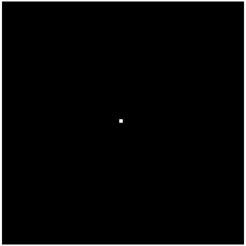
$$J(x,y) = (I * H)(x,y) = \sum_i \sum_j H(i,j) I(x-i, y-j)$$

Pour des raisons de simplicité, le masque H est :

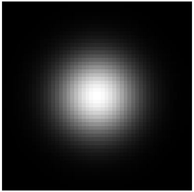
→ centré (pixel $(0,0)$ situé au milieu du masque) ;

→ de taille impaire (3×3, 5×5, 7×7, ...).

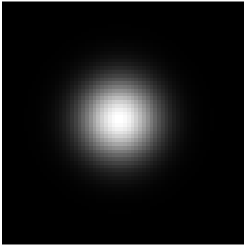
Exemple



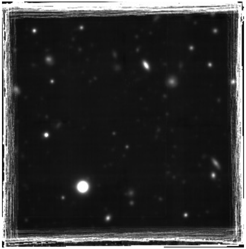
$*$



$=$

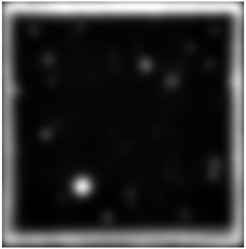


Exemple – Point spread function (PSF)



$*$

$=$



Exemple – Filtre gaussien

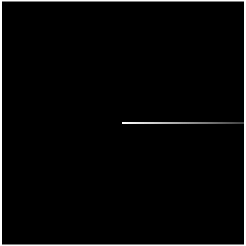
 $*$  $=$ 

Exemple – Flou de bougé

 $*$  $=$ 

Exemple – Flou de bougé

$*$



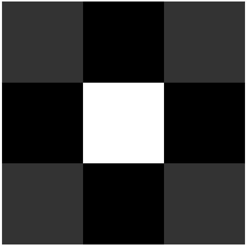
$=$

Exemple – Filtre passe-haut

$*$ $\cdot$ $=$

Exemple – Filtre passe-haut

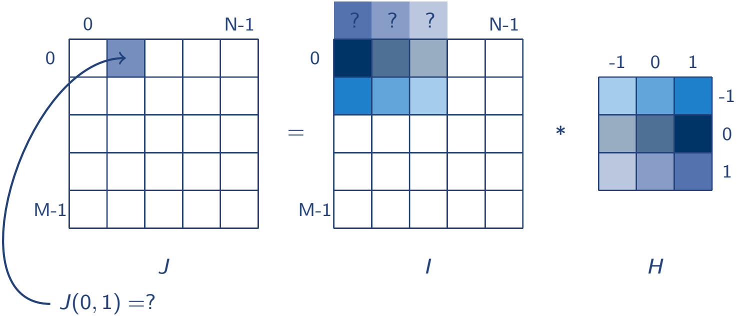
$*$



$=$

Problèmes aux bords

$$J(x,y) = (I * H)(x,y) = \sum_i \sum_j H(i,j) I(x-i, y-j)$$



$\Rightarrow$ les pixels en dehors de l'image doivent être fixés : plusieurs manières possibles.

Problèmes aux bords – Agrandir l'image

Périodisation

Complétion avec des zéros

Reproduire le bord

Effectuer un miroir

Problèmes aux bords – Agrandir l'image

Périodisation

Complétion avec des zéros

Reproduire le bord

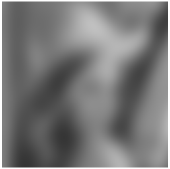
Effectuer un miroir

Problèmes aux bords – Troncature du résultat

Le résultat peut être tronqué (cf. conv2 en Matlab) :

'full'

'same'



'valid'

Commande Matlab

```
>> [N,M] = size(I_originale);
>> [P,Q] = size(filtre);
>> Iconv = conv2(I_originale, filtre, 'full');
>> size(Iconv) = [N + P-1, M + Q-1];
>> Iconv = conv2(I_originale, filtre, 'same');
>> size(Iconv) = [N, M];
>> Iconv = conv2(I_originale, filtre, 'valid');
>> size(Iconv) = [N - (P-1), M - (Q-1)];
```

Problèmes aux bords

Il n'y a pas de méthode parfaite : toutes introduisent des erreurs.

$\Rightarrow$ s'arranger pour que les objets d'intérêt soient loin du bord.

Propriétés

→ Élément neutre : image nulle avec un seul pixel égal à 1

→ Commutativité : $I * H = H * I$

→ Distributivité : $I * (H_1 + H_2) = I * H_1 + I * H_2$

→ Linéarité ($\alpha \in \mathbb{C}$) : $\alpha(I * H) = (\alpha I) * H = I * (\alpha H)$

→ Associativité : $H_1 * (H_2 * H_3) = (H_1 * H_2) * H_3$

Séparabilité

Les filtres H pouvant s'écrire comme la convolution de deux filtres 1D suivant les deux axes (H_x et H_y) sont appelés **filtres séparables**.

$$\underbrace{\begin{bmatrix} a & b & c \end{bmatrix}}_{H_x} * \underbrace{\begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix}}_{H_y} = \underbrace{\begin{bmatrix} a\alpha & b\alpha & c\alpha \\ a\beta & b\beta & c\beta \\ a\gamma & b\gamma & c\gamma \end{bmatrix}}_{H}$$

Existe-t-il un opérateur inverse de la convolution ?

→ Ce problème est appelé déconvolution.

→ Si la PSF est connue et vérifie certaines conditions très particulières (cf. analyse de Fourier) : c'est possible !

→ En pratique, la quantification et le bruit rendent la déconvolution difficile.

Plan du chapitre

Joseph Fourier

1768–1830

Géomètre et physicien

Égyptologue

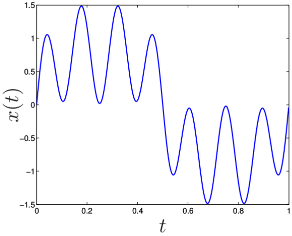
Préfet d'Isère

Professeur à Polytechnique

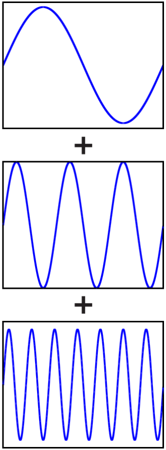
Membre de l'Académie des sciences

Transformée de Fourier d'un signal

Tout signal peut s'écrire comme une somme de sinusoïdes :

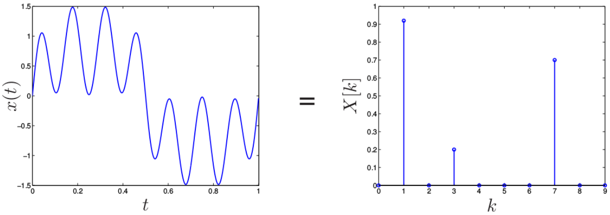


=



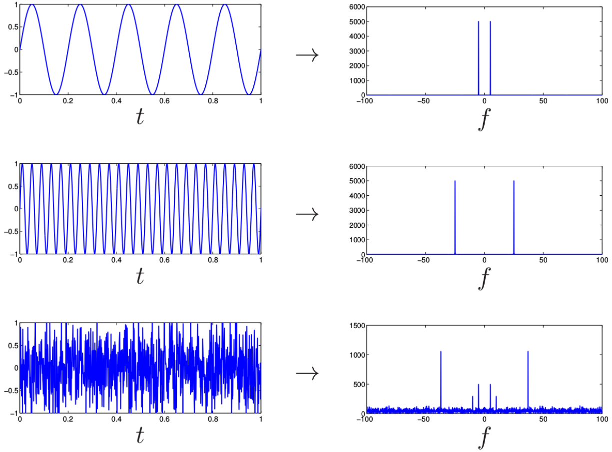
Transformée de Fourier d'un signal

Tout signal peut s'écrire comme une somme de sinusoïdes :



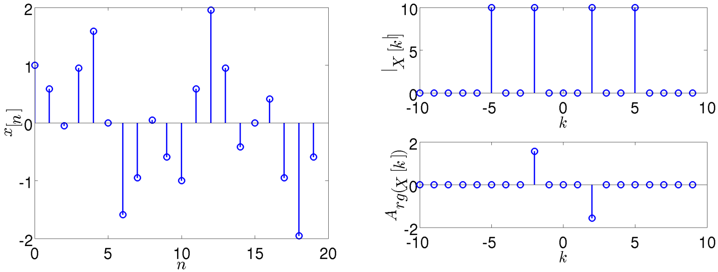
Transformée de Fourier d'un signal

La TF fait apparaître les fréquences contenues dans un signal :



Transformée de Fourier discrète 1D

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi kn} \qquad x[n] = \sum_{k=0}^{N-1} X[k] e^{+j2\pi kn}$$



Transformée de Fourier bidimensionnelle

Transformée de Fourier

La transformée de Fourier discrète d'une image de taille $M \times N$ est :

$$\mathcal{I}(u,v) = \frac{1}{MN}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} I(i,j)e^{-j\,2\pi\left(\frac{ui}{M}+\frac{vj}{N}\right)}$$

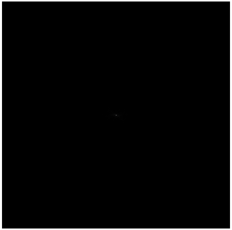
La transformée de Fourier est donc elle-même une image de taille $M \times N$, à valeurs complexes.

Transformée de Fourier inverse

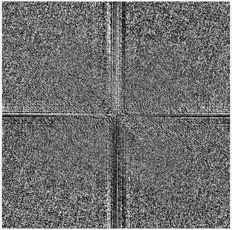
La transformée de Fourier inverse discrète d'une image de taille $M \times N$ est :

$$I(i,j) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} \mathcal{I}(u,v)e^{+j\,2\pi\left(\frac{ui}{M}+\frac{vj}{N}\right)}$$

Transformée de Fourier de Lena

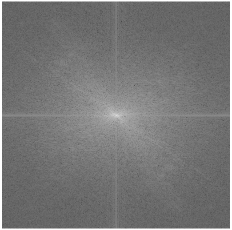


Module

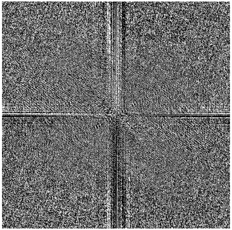


Phase

Transformée de Fourier de Lena



Module (échelle log)



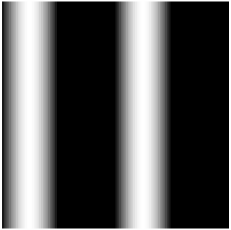
Phase

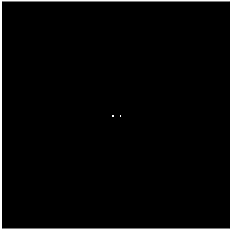
Fréquences dans une image

Hautes fréquences

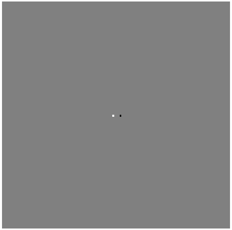
Basses fréquences

Transformée de Fourier bidimensionnelle



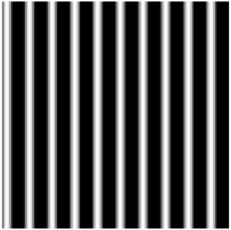


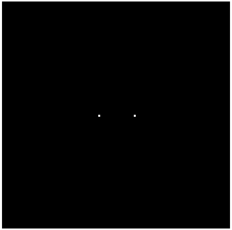
Module



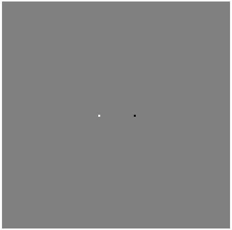
Phase

Transformée de Fourier bidimensionnelle





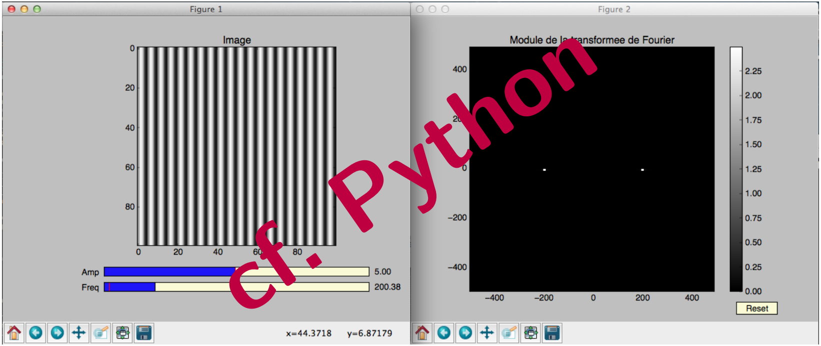
Module



Phase

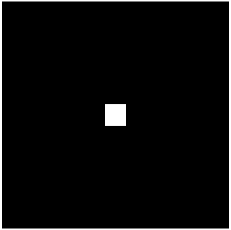
Transformée de Fourier bidimensionnelle

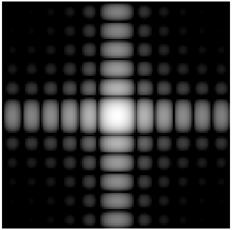
→ Attention aux problèmes d'échantillonnage (repliement, Shannon doit être aussi vérifié en 2D).



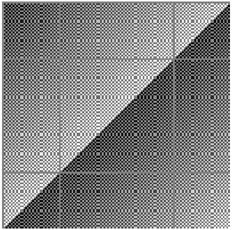
cf TF2D.py

Transformée de Fourier bidimensionnelle



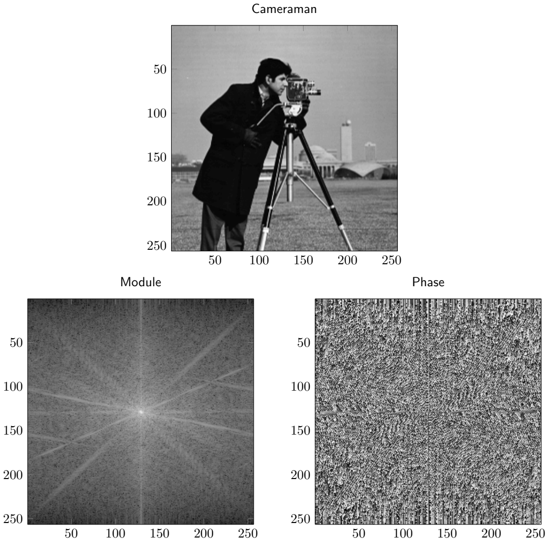


Module



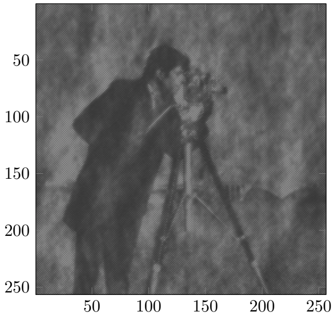
Phase

Transformée de Fourier bidimensionnelle

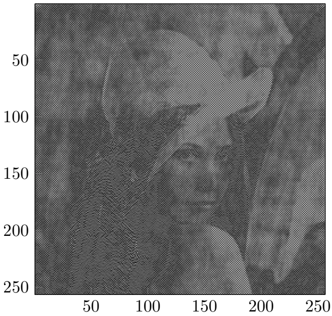


Transformée de Fourier bidimensionnelle

Module Lena + Phase Cameraman



Module Cameraman + Phase Lena



→ Amplitude : indique seulement quelle structure périodique est contenue dans l'image, mais pas où.

→ Phase : informations essentielles sur la structure de l'image.

Plan du chapitre

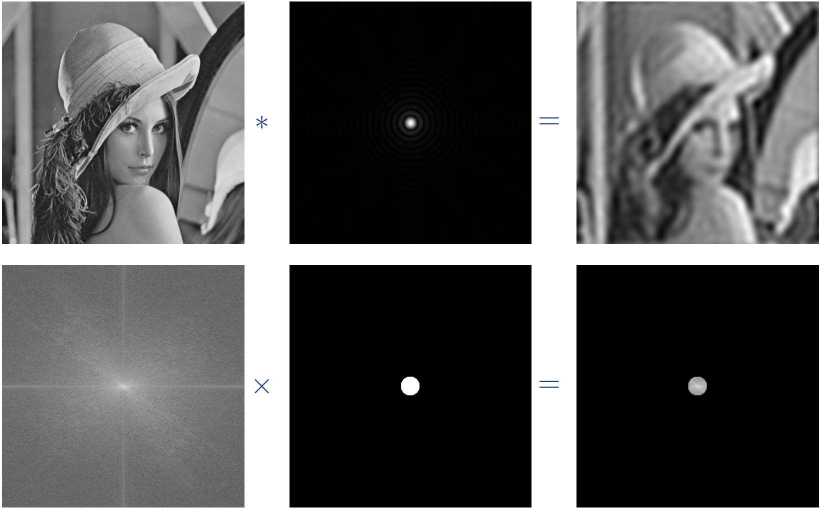
Filtrage

De même que pour les signaux 1D, le filtrage (convolution 2D) peut se faire dans le domaine fréquentiel (multiplication élément par élément) :

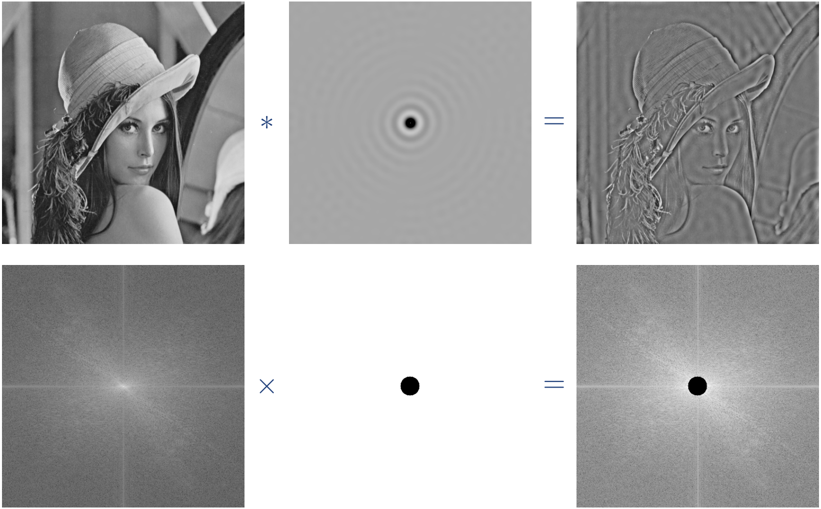
$$I_1 * I_2 \Leftrightarrow TF^{-1}(\mathcal{I}_1 \times \mathcal{I}_2)$$

où $\mathcal{I}_1$ (resp. $\mathcal{I}_2$) est la TF 2D de l'image I_1 (resp. I_2).

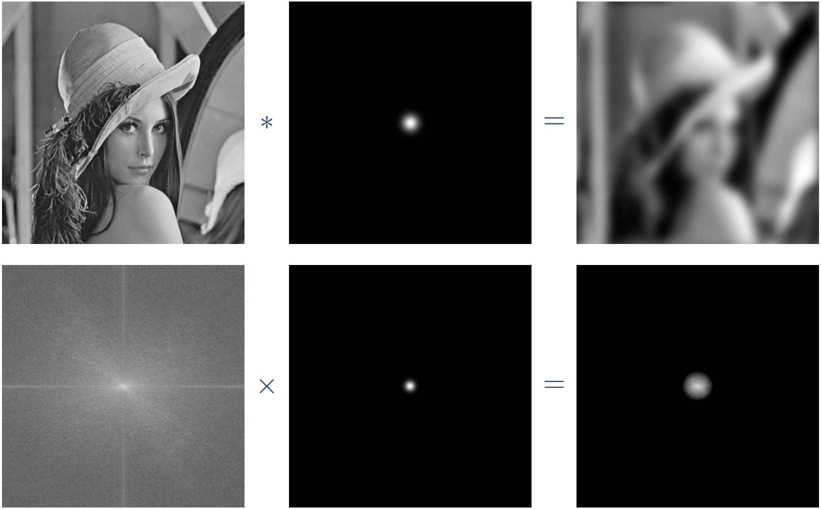
Filtrage



Filtrage



Filtrage



A suivre ...

Restauration d'images