

## TP10 – Décomposition *cartoon+texture* d'une image

Une des multiples applications de la transformation de Fourier (TF) est le *filtrage spectral*. Soit un signal unidimensionnel  $f(t) : \mathbb{R} \rightarrow \mathbb{R}$ . On appelle *filtre spectral* toute application qui, au *spectre d'entrée*  $F_e(\nu) : \mathbb{R} \rightarrow \mathbb{C}$  de  $f(t)$ , obtenu par TF, associe un *spectre de sortie*  $F_s(\nu) : \mathbb{R} \rightarrow \mathbb{C}$  tel que  $\forall \nu \in \mathbb{R}, |F_s(\nu)| \leq |F_e(\nu)|$ . Les filtres les plus connus sont les *passé-bas* et les *passé-haut*. Les filtres des figures 1-a et 1-b, qui sont caractérisés par la même *fréquence de coupure*, sont complémentaires : la somme des deux spectres de sortie est égale au spectre d'entrée. On peut construire des filtres plus sophistiqués en combinant judicieusement un filtre passe-bas et un filtre passe-haut de fréquences de coupure différentes : un filtre *passé-bande* s'obtient en faisant subir au spectre d'entrée un filtrage passe-bas puis un filtrage passe-haut (cf. figure 1-c) ; un filtre *coupe-bande* s'obtient en sommant les spectres de sortie résultant d'un filtrage passe-bas et d'un filtrage passe-haut (cf. figure 1-d).

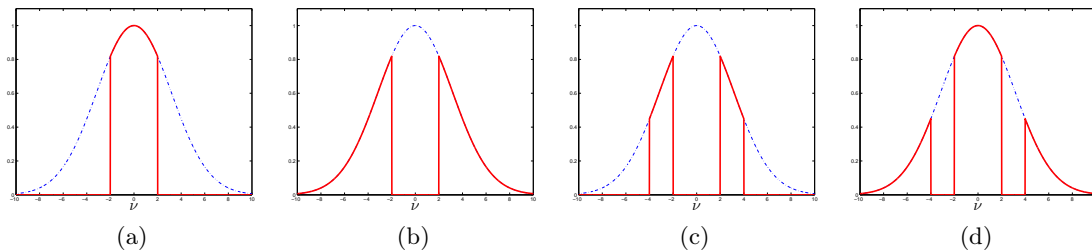


FIGURE 1 – Exemples de filtres : (a) passe-bas, (b) passe-haut, (c) passe-bande et (d) coupe-bande.

### Exercice 1 : filtrage spectral

Le script `exercice_0.m` lit une image, calcule son spectre, c'est-à-dire sa transformée de Fourier discrète bidimensionnelle (fonction `fft2` de Matlab), puis décompose ce spectre en deux parties, grâce à un filtre passe-bas  $\Phi_b$  et à un filtre passe-haut  $\Phi_h$  complémentaires :

$$\begin{cases} \Phi_b(\nu_x, \nu_y) = \chi(\sqrt{\nu_x^2 + \nu_y^2} \leq \nu_c) \\ \Phi_h(\nu_x, \nu_y) = \chi(\sqrt{\nu_x^2 + \nu_y^2} > \nu_c) \end{cases}$$

où  $\chi(s)$  vaut 1 si  $s$  est vrai et 0 sinon (fonction caractéristique), et où  $\nu_c$  désigne la fréquence de coupure. Les images correspondant aux basses fréquences et aux hautes fréquences sont obtenues par transformation de Fourier discrète inverse bidimensionnelle (fonction `ifft2` de Matlab), dont seule la partie réelle est conservée.

Ce script permet d'effectuer une séparation grossière de l'image en une partie « plutôt photométrique » et une partie « plutôt géométrique », connue sous le nom de *décomposition cartoon+texture*. Testez différentes valeurs de  $\nu_c$  afin d'obtenir une décomposition « optimale », le critère d'optimalité étant très subjectif.

Même en choisissant la valeur « optimale » de  $\nu_c$ , le résultat du script `exercice_0.m` reste visuellement médiocre. Cela vient de la coupure franche entre basses fréquences et hautes fréquences. Pour améliorer l'extraction des basses fréquences, une solution consiste à pondérer le spectre non pas par une fonction caractéristique, mais par une fonction lisse qui soit décroissante lorsque la distance à l'origine (dans le plan de Fourier) croît.

Faites une copie de `exercice_0.m`, de nom `exercice_1.m`, afin d'utiliser le filtre passe-bas suivant :

$$\Phi_b(\nu_x, \nu_y) = \frac{1}{1 + \frac{\nu_x^2 + \nu_y^2}{\eta}} \quad (1)$$

et le filtre passe-haut complémentaire  $\Phi_h(\nu_x, \nu_y) = 1 - \Phi_b(\nu_x, \nu_y)$ , et donnez au paramètre  $\eta$  la valeur 0,05.

## Exercice 2 : modèle ROF

Le problème de la décomposition *cartoon+texture* peut être résolu sans passer explicitement dans le domaine de Fourier. La décomposition d'une image  $u$  en deux parties  $c$  (*cartoon*, qui correspond aux basses fréquences) et  $t$  (*texture*, qui correspond aux hautes fréquences) est telle que  $c + t = u$ , si les filtres passe-bas et passe-haut utilisés sont complémentaires, car la transformation de Fourier est linéaire. L'image  $t$  se déduit donc très simplement de l'image  $c$ . Par ailleurs, nous attendons de  $c$  qu'elle soit « proche » de l'image originale  $u$  et que les variations de niveau de gris soient moins brutales dans  $c$  que dans  $u$ . Nous pouvons donc utiliser le *modèle ROF* (proposé par Rudin, Osher et Fatemi en 1992), qui fait apparaître la variation totale :

$$E_{\text{ROF}}(c) = \iint_{\Omega} \left\{ \frac{1}{2} [c(x, y) - u(x, y)]^2 + \lambda |\nabla c(x, y)|_{\epsilon} \right\} dx dy \quad (2)$$

où  $\lambda > 0$  est un paramètre du modèle et  $|\nabla c(x, y)|_{\epsilon} = \sqrt{|\nabla c(x, y)|^2 + \epsilon}$ . Ce modèle a déjà été rencontré dans l'exercice 1 du TP7 (modèle de débruitage par variation totale). L'équation d'Euler-Lagrange associée à la fonctionnelle  $E_{\text{ROF}}(c)$  est non linéaire. Elle peut être résolue par un schéma itératif de type « point fixe », qui consiste à calculer la solution à l'itération  $(k+1)$  en « figeant » la partie non linéaire à l'itération  $(k)$  :

$$\left[ I - \lambda \nabla \cdot \left( \frac{1}{\sqrt{|\nabla c^{(k)}(x, y)|^2 + \epsilon}} \nabla \right) \right] c^{(k+1)}(x, y) = u(x, y), \quad \forall (x, y) \in \Omega \quad (3)$$

où l'opérateur  $\nabla \cdot$  désigne la *divergence*. Après discrétisation, l'itération (3) s'écrit :

$$\left[ \mathbf{I}_N - \lambda \left( -\mathbf{D}_x^{\top} \mathbf{W}^{(k)} \mathbf{D}_x - \mathbf{D}_y^{\top} \mathbf{W}^{(k)} \mathbf{D}_y \right) \right] \mathbf{c}^{(k+1)} = \mathbf{u} \quad (4)$$

Dans (4),  $N$  désigne le nombre de pixels de  $\Omega$ ,  $\mathbf{I}_N$  est la matrice identité, et  $\mathbf{W}^{(k)}$  la matrice diagonale des coefficients  $\frac{1}{\sqrt{|\nabla c^{(k)}(x, y)|^2 + \epsilon}}$  calculés en tous les pixels de  $\Omega$ . Ces deux matrices de taille  $N \times N$  peuvent être calculées à l'aide des fonctions `speye` et `spdiags`, respectivement. Les vecteurs  $\mathbf{c}^{(k+1)}$  et  $\mathbf{u}$  concatènent les valeurs de  $c^{(k+1)}(x, y)$  et  $u(x, y)$  calculées en tous les pixels de  $\Omega$ . Quant aux matrices  $\mathbf{D}_x$  et  $\mathbf{D}_y$ , elles permettent de calculer par *différences finies* les dérivées partielles d'une fonction  $f : \Omega \rightarrow \mathbb{R}$ , pour peu que  $\Omega$  soit rectangulaire. Ces deux matrices bi-diagonales de taille  $N \times N$  se calculent aussi à l'aide de la fonction `spdiags` (cf. TP7).

La résolution de l'équation (4), que l'on réécrit  $\mathbf{A}^{(k)} \mathbf{c}^{(k+1)} = \mathbf{u}$ , peut être effectuée très simplement à l'aide de l'opérateur `\` de Matlab. Enfin, il est naturel de choisir comme initialisation de cette itération  $\mathbf{c}^{(0)} = \mathbf{u}$ .

Faites une copie du script `exercice_1.m` du TP7, de nom `exercice_2.m`, que vous modifierez de manière à effectuer 20 itérations du schéma numérique (4), avec  $\lambda = 50$  et  $\epsilon = 0,01$ . Un fois mis au point, faites une copie de ce script, sous le nom `exercice_2_bis.m`, que vous modifierez de manière à lire l'image `empreinte.png`, et à utiliser  $\lambda = 5$ . En seuillant à 2 la texture  $t$  obtenue au fil des itérations, vous obtiendrez une image binaire de l'empreinte digitale, débarrassée des variations « douces » du niveau de gris. Pour valider l'intérêt de cette méthode d'extraction de la texture, seuillez l'image originale  $u$  et comparez les résultats.

## Exercice 3 : modèle TV-Hilbert

Il est également possible de résoudre le problème de la décomposition *cartoon+texture* d'une image en utilisant le *modèle TV-Hilbert*, qui est un modèle mixte, dans lequel un des deux termes dépend des spectres  $\hat{c}$  et  $\hat{u}$  de l'inconnue  $c$  et de la donnée  $u$ , tandis que l'autre terme dépend de l'inconnue  $c$  :

$$E_{\text{TV-H}}(c) = \frac{1}{2} \iint_{\Omega} \{ \Phi_b(\nu_x, \nu_y) |\hat{c}(\nu_x, \nu_y) - \hat{u}(\nu_x, \nu_y)|^2 \} d\nu_x d\nu_y + \lambda \iint_{\Omega} |\nabla c(x, y)|_{\epsilon} dx dy \quad (5)$$

Le filtre passe-bas  $\Phi_b(\nu_x, \nu_y)$  du modèle (5) a déjà été défini en (1). Le terme d'*attache aux données* semble effectivement mieux adapté au problème que celui de  $E_{\text{ROF}}(c)$ , car seules les basses fréquences des images  $u$  et  $c$  sont contraintes à être égales. Un calcul un peu plus compliqué que celui de l'exercice 2 fournit l'équation d'Euler-Lagrange suivante :

$$\text{TF}^{-1} \{ \Phi_b [\hat{c} - \hat{u}] \} (x, y) - \lambda \nabla \cdot \left( \frac{\nabla c(x, y)}{|\nabla c(x, y)|_{\epsilon}} \right) = 0, \quad \forall (x, y) \in \Omega \quad (6)$$

Pour résoudre l'équation (6), nous utilisons un schéma de *descente de gradient* qui consiste à appliquer l'itération suivante (le *pas de descente*  $\gamma > 0$  doit être fixé par l'utilisateur) :

$$c^{(k+1)}(x, y) = c^{(k)}(x, y) - \gamma \left[ \text{TF}^{-1} \left\{ \Phi_b [\hat{c}^{(k)} - \hat{u}] \right\} (x, y) - \lambda \nabla \cdot \left( \frac{\nabla c^{(k)}(x, y)}{|\nabla c^{(k)}(x, y)|_\epsilon} \right) \right], \quad \forall (x, y) \in \Omega \quad (7)$$

Bien qu'elle soit réelle en théorie, les valeurs retournées par la fonction  $\text{TF}^{-1} \left\{ \Phi_b [\hat{c}^{(k)} - \hat{u}] \right\}$  comportent, en pratique, une petite partie imaginaire qu'il convient d'éliminer en ne conservant que la partie réelle.

Un calcul fastidieux, mais sans difficulté, permet d'explicitier le dernier terme du membre droit de (7) :

$$\nabla \cdot \left( \frac{\nabla c}{|\nabla c|_\epsilon} \right) = \frac{c_{xx}(c_y^2 + \epsilon) + c_{yy}(c_x^2 + \epsilon) - 2c_x c_y c_{xy}}{(c_x^2 + c_y^2 + \epsilon)^{3/2}} \quad (8)$$

où  $c_x$ ,  $c_y$ ,  $c_{xx}$ ,  $c_{xy}$  et  $c_{yy}$  désignent les dérivées partielles d'ordre 1 et 2 de la fonction  $c$ . Étant donné que le domaine  $\Omega$  est rectangulaire, puisqu'il contient la totalité de l'image, ces dérivées partielles peuvent être calculées comme dans l'exercice 2, en utilisant les matrices  $\mathbf{D}_x$  et  $\mathbf{D}_y$  :

$$\mathbf{c}_x = \mathbf{D}_x \mathbf{c} \quad ; \quad \mathbf{c}_y = \mathbf{D}_y \mathbf{c} \quad ; \quad \mathbf{c}_{xx} = -\mathbf{D}_x^\top \mathbf{D}_x \mathbf{c} \quad ; \quad \mathbf{c}_{xy} = -\mathbf{D}_x^\top \mathbf{D}_y \mathbf{c} \quad ; \quad \mathbf{c}_{yy} = -\mathbf{D}_y^\top \mathbf{D}_y \mathbf{c}$$

Il est rappelé que, dans ces expressions,  $\mathbf{c}$ ,  $\mathbf{c}_x$ ,  $\mathbf{c}_y$ ,  $\mathbf{c}_{xx}$ ,  $\mathbf{c}_{xy}$  et  $\mathbf{c}_{yy}$  désignent des matrices vectorisées.

Écrivez un script `exercice_3.m` permettant d'effectuer 1000 itérations du schéma numérique découlant de la discrétisation de (7), avec  $\lambda = 1000$ ,  $\epsilon = 0,5$  et  $\gamma = 0,0001$ . Pour ne pas trop ralentir l'exécution de ce script, il est conseillé de n'effectuer la mise à jour des affichages que toutes les 20 itérations.