UNIVERSITÉ PARIS-EST MARNE-LA-VALLÉE

Travaux dirigés Traitement d'images n°5

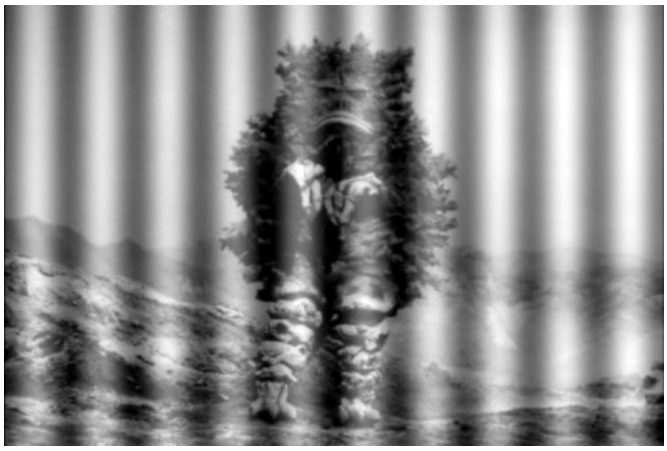
Transformée de Fourier

—Master 1—

▸ Exercice 1. *Fourier : premier contact*

1. *Ouvrez le fichier* `tp05/tp05_ex01.cpp`*, il génère la transformée de Fourier d'une image, puis effectue la transformée inverse. Essayez sur une image.*

2. *L'image* `rayures.png` *semble bruité. Trouvez une solution pour la débarasser de ses rayures. À savoir, les images de magnitude et de phase contiennent des* `float` *et non des* `unsigned char`*, on accède donc au pixel (i, j) de ces images par* `image.at<float>(i,j)`*.*



▸ Exercice 2. *Fourier : filtres fréquentiels*

1. *Trouvez une belle image sur internet.*

2. *Modifiez la magnitude de la transformée de Fourier de l'image pour obtenir un filtre passe bas : seules les variations globales de l'image apparaissent. Vous pouvez éventuellement utiliser la fonction* `removeRing` *du fichier* `fourierTransform.hpp`*.*

3. *Modifiez la magnitude de la transformée de Fourier de l'image pour obtenir un filtre passe haut : seuls les détails de l'image apparaissent.*

4. *Modifiez la magnitude de la transformée de Fourier de l'image pour obtenir un filtre passe bande : seule une bande de fréquences de l'image apparait.*

▸ **Exercice 3.** ***Fourier : filtres de convolution***

1. *Voici à quoi ressemble le filtre de Sobel 3×3 sur l'axe des x :*

$$\mathbf{S}_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Faites une fonction `cv::Mat sobelXFourierKernel(const int width, const int height)` *qui génère une image de* `float` *de taille* `height`×`width` *(taille de l'image à traiter), constituée de zeros. Vous pouvez utiliser le constructeur suivant pour faire une image de* `float` *pleine de 0 :*
`cv::Mat kernel(height, width, CV_32F, cv::Scalar(0.0))`.

Remplacer les 9 zeros du centre de l'image par les valeurs du filtre $\mathbf{S}_x$.

2. *Calculez les transformées de Fourier de votre image et du filtre. Multipliez chaque terme (i, j) de la magnitude de l'image par le terme (i, j) de la magnitude du filtre. Faites enfin la transformée de Fourier inverse de votre image avec la nouvelle magnitude précédement modifiée. Que constatez-vous?*

3. *A quoi peut-bien ressembler le filtre de Sobel 3×3 sur l'axe des y? Faites une fonction* `cv::Mat sobelYFourierKernel(const int width, const int height)` *et appliquez ce filtre à l'image dans le domaine de Fourier.*

4. *Faites de même avec un filtre Laplacien :*

$$\mathbf{L} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

5. *et avec un réhausseur de contours (à tester sur l'image de Garamon) :*

$$\mathbf{R} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

6. *Faites une fonction* `cv::Mat gaussianFourierKernel(const int width, const int height, const double sigma)` *qui génère un filtre Gaussien. Celui-ci peut tout à fait avoir la taille de l'image. Pour rappel, une gaussienne peut s'écrire :*

$$g(x, y) = \frac{1}{2\pi\sigma^2} \exp\Big(- \frac{x^2 + y^2}{2\sigma^2} \Big)$$

où x et y sont exprimés dans dans un repère dont l'origine est au centre du noyau. Pensez à bien normaliser les coefficients de votre filtre de telle sorte que leur somme vaut 1 : le coéfficient $\frac{1}{2\pi\sigma^2}$ n'est valide que pour un noyaux de taille infinie.
Appliquez ce filtre à l'image dans le domaine de Fourier. Refaites la même expérience avec un filtre Gaussien 3×3 redimensioné à la taille de l'image. Constatez-vous une différence?

7. *Faites votre propre filtre.*