TP4 - Hiver 2018

IMN 259

Analyse d'images

Date limite pour remettre votre travail : 16 avril

Objectifs

Implémenter différents filtres dont voici la liste :

1. Filtrage spatial passe-bas
   (a) Filtre passe-bas Gaussien
   (b) Filtre moyenneur
2. Filtrage spatial passe-haut
   (a) Filtre passe-haut Gaussien
3. Filtre médian
4. Corrélation
5. Diffusion non-linéaire
6. Introduction à la transformée de Fourier (TF)
   (a) Filtres passe-bas et passe-haut
   (b) Filtre moyenneur spectral

Description

À l'aide du code C++ fourni (fichiers tp4A.cpp, tp4B.cpp, tp4C.cpp, tp4D.cpp, tp4E.cpp, tp4F.cpp, tp4G.cpp, tp4H.cpp, tp4I.cpp, MImage.h, MImage.cpp) vous devez implémenter différents filtres vus dans le cours (un par fichier tp4X.cpp). Pour ce faire, il est fortement recommandé de récupérer les fonctions *MedianFilter*, *LowpassGaussianFilter*, *CorrelationFilter*, *NonLinearDiffusionFilter*, *AverageFilter*, *RescaledCorrelationFilter*, *HighpassGaussianFilter*, *SpectralRectLowPassFilter* et *SpectralRectHighPassFilter* de la classe *MImage*. Toutefois, vous être libre d'ajouter d'autres fonctions à la classe MImage si vous en éprouvez le besoin.

Recommandations pour les filtres :

1. tp4A : **Filtre moyenneur**. Calculer, pour chaque pixel, l'intensité moyenne locale à l'intérieur d'un voisinage de taille : (2 halfSize +1)×(2 halfSize +1). La variable "halfSize" est la demi-taille du masque. Elle est passée en paramètre à la fonction "AverageFilter".
2. tp4B : **Filtre passe-bas gaussien**. Convoluer l'image d'entrée à l'aide d'un filtre gaussien de taille $(6\sigma + 1) \times (6\sigma + 1)$. En guise de rappel, les valeurs du masque gaussien doivent suivre la distribution suivante :

$$\frac{1}{2\pi\sigma^2} \exp(-((x - x_0)^2 + (y - y_0)^2)/2\sigma^2) \tag{1}$$

   où σ est l'écart-type et (x_0, y_0) est le centre du masque. Par exemple, lorsque $\sigma = 1$, le masque doit avoir une taille de 7×7 et $(x_0, y_0) = (3, 3)$.

3. tp4C : **Filtre passe-haut gaussien**. Ce filtre s'implémente de façon similaire au filtre passe-bas gaussien. La différence est que le masque doit contenir les valeurs de la distribution suivante :

$$\delta(x - x_0, y - y_0) - \frac{1}{2\pi\sigma^2} \exp(-((x - x_0)^2 + (y - y_0)^2)/2\sigma^2) \tag{2}$$

où $\delta(x - x_0, y - y_0)$ est un delta de Dirac placé au centre du masque. À noter que le masque de ce filtre doit avoir la taille $(6\sigma + 1) \times (6\sigma + 1)$.

4. tp4D : **Corrélation**. Appliquer l'opération de corrélation entre les images "letters" et "correlationImage" :

$$f \circ h = \sum_s \sum_t f(s,t)h(x + s, y + t) \tag{3}$$

Pour aider à la visualisation, recalez l'image résultante entre 0 et 255.

5. tp4E : **Corrélation recalée**. Appliquer l'opération de corrélation recalée entre deux images f et h

$$(f \circ h)_{Recal} = \sum_s \sum_t (f(s,t) - \bar{f}(s,t))(h(x + s, y + t) - \bar{h}) \tag{4}$$

où $\bar{f}(s,t)$ est la moyenne locale de l'image f dans la zone couverte par h et $\bar{h}$ est la moyenne globale de h. Pour aider à la visualisation, recalez l'image résultante entre 0 et 255 à l'aide de la fonction "Rescale". Vous pouvez tester votre code avec les images "barbara.pgm" et "nose.pgm" (en niveaux de gris) ou encore les images "olives.ppm" et "corrOlive.ppm" (en couleur).

6. tp4F : **Filtre médian**. Débruiter une image poivre et sel à l'aide d'un filtre médian spatial. Il est fortement recommandé d'utiliser un algorithme de tri déjà existant comme la commande "qsort" ou encore `std::sort`. Pour plus d'information sur ces commandes, visitez les sites web :

http ://www.cplusplus.com/reference/clibrary/cstdlib/qsort.html
http ://www.cplusplus.com/reference/algorithm/sort/
http ://www.fredosaurus.com/notes-cpp/algorithms/sorting/stl-sort-arrays.html.

7. tp4G : **Diffusion non linéaire**. Implémenter l'algorithme de diffusion non linéaire de type *explicite* vu dans le cours (algorithme C1). Pour ce faire, vous utiliserez la fonction de diffusivité de Perona-Malik :

$$g(|\nabla u|) = \frac{1}{1 + \frac{|\nabla u|^2}{\lambda^2}} \tag{5}$$

où $|\nabla u| = \sqrt{(\partial u/\partial x)^2 + (\partial u/\partial y)^2}$.

8. tp4H : **Filtrage fréquentiel passe bas et passe haut.** Vous devez prendre la TF d'une image et lui appliquer un filtre fréquentiel passe-bas (et passe-haut) rectangulaire (fonction porte). Pour ce faire, il vous faut compléter les fonctions "SpectralRectLowPassFilter" et "SpectralRectHighPassFilter". Si vous le désirez, vous pouvez ajouter un (ou plusieurs) recalage cyclique dans la fonction "main".

9. tp4I : **Filtre moyenneur spectral**. Au tp4A, vous avez implémenté un filtre moyenneur **spatial**. Vous avez donc implémenté une fonction vous permettant de convoluer une image à l'aide d'une fonction porte. Vous devez ici implémenter cette même opération, mais cette fois-ci via le domaine spectral. Cela vous permettra de constater qu'une multiplication dans le domaine spectral prend moins de temps qu'une convolution dans le domaine spatial lorsque le filtre est de grande taille.

   Pour y arriver, vous devez d'abord ouvrir une image d'entrée $f(x, y)$ de taille $N \times M$ (par exemple, le caméraman, de taille 256×256). Pour filtrer $f(x, y)$ dans le domaine spectral, vous devez créer une seconde image de taille $N \times M$ qu'on appellera $h(x, y)$. Ici, $h(x, y)$ est une « image filtre » comprenant les valeurs d'un filtre moyenneur. Par exemple, si $h(x, y)$ est un filtre moyenneur 3×3 du type

$$\left| \begin{array}{ccc} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{array} \right| /9,$$

   l'image $h(x, y)$ contiendra une valeur de $1/9$ dans les 9 pixels situés autour de l'origine et des zéros partout ailleurs. Une fois $f(x, y)$ et $h(x, y)$ connues, il ne reste plus qu'à prendre leur transformée de Fourier, les multiplier ensemble et prendre la transformée de Fourier inverse du signal résultant. En d'autres mots,

$$\begin{array}{lll} 1. & H(u, v) & = \mathcal{F}(h(x, y)) \\ 2. & F(u, v) & = \mathcal{F}(f(x, y)) \\ 3. & G(u, v) & = F \times H \\ 4. & g(x, y) & = \mathcal{F}^{-1}(G(u, v)) \\ 5. & \text{Sauvegarder} & g(x, y) \end{array} \tag{6}$$

Il est à noter que tous ces filtres doivent fonctionner autant pour des images en niveaux de gris que pour des images en couleur. Pour les images couleur, vous devez filtrer chacune des bandes individuellement. Cette règle ne s'applique toutefois pas aux programmes "tp4G", "tp4H" et "tp4I". Aussi,vous pouvez retenir l'option de votre choix pour gérer les bords lors de filtrages spatiaux (enroulement, réflexion, étirement, ajout de zéros).

Évaluation

Ce travail doit être fait en **équipe de DEUX**. Pour simplifier la correction, nous vous demandons de modifier le moins possible les fonctions **main** et de remettre TOUS les fichiers incluant le Makefile, les images et les fichier tp4X.cpp. Au moment de soumettre votre travail, assurez-vous que votre code compile bien sous Linux (vous n'avez qu'a taper la commande *make* dans un terminal Linux). Utilisez le turnin web sur le site http ://opus.dinf.usherbrooke.ca :8080/ pour soumettre votre travail. À noter qu'une pénalité de 10% par jour sera appliquée à tout travail remis en retard.