1 Introduction

Rappel... toujours les meme Avant tout, nous vous rappelons, même si ce n'est pas nécessaire, que la documentation et des exemples consernant les fonctions que vous allez utiliser est disponnible soit dans le logiciel via la commande *help* soit en cherchant sur le web via votre moteur de recherche préférer.

1.1 À rendre

De votre part, nous attendons deux éléments :

- Un compte rendu expliquant votre démarche, vos codes : il faut que vous montriez que vous avez compris de quoi il s'agit de faon concise (8 pages maximum sans les figures)
- Des script : un pour chaque question avec des figures si nécessaire (toujours commentées)

vous disposez d'une semaine pour rendre votre TP sous format électronique. Un depot sera ouvert sur le e-campus a partir du milieu de la semaine. Vous devrez y deposer une archive (zip ou tar) contenant :

- deux dossiers : un dossier contenant votre résumé et vos figures, un autre contenant vos script
- les script doivent être clairement identifiable. On doit pouvoir savoir de quelle question il s'agit dans le nom des fichiers ".m".
- Les script doivent montrer les éléments de réponse que ce soit en figure ou en chiffre. Chaque morceau de code doit être commenté lorsqu'il s'agit d'une opération "importante".
- Votre résumé (au format pdf), qui ne doit pas nécessairement être long (on ne note pas au kilo), doit comporter les éléments de bases pour être identifier : noms et prénoms du binôme ainsi que le nom du TP. SI vous faite référence a du cours, il faut écrire de faon complète la propriété (pas de cf...). Les figures ne sont pas des justifications par elles meêmes. Elles doivent être commentées et expliques. Nous vous conseillons de faire une petit résumé par question comportant objectif, méthode, résultat et conclusion (cela peut être aussi court que deux ou trois phrases).
- Pour tout autres questions, nous somme disposes à vous répondre soit par mail ou a nos bureau respectifs.
- Vous disposez de 10 jours après le TP pour nous rendre votre compte rendu. Nous nous attendons à ce que vous aillez le temps de faire les parties 2 et 3 au cours du TP, cependant, il faut traiter toutes les parties, en particulier la section 4.6 (reconstruction par marqueurs) que nous vous suggérons de faire après la section 3 (du point de vue temporel).

1.2 Les moyens

Vous trouverez toutes les ressources l'adresse suivante : `https://github.com/matthewozon/TPfiltrage4ETI`
Consigne à observer Le TP doit se dérouler sous OCTAVE que vous pouvez lancer soit en mode terminal via la commande *octave* soit en démarrant l'interface graphique *QtOctave*.

conseil pour les figures Utiliser la commande *print('-depsc','nom_du_fichier')* qui "imprime" ce que vous voyez dans la figure sans en diminuer la qualité (contrairement au capture décran). Pour plus d'information sur cette commande, nous vous invitons a taper dans votre prompt octave *help print*.

Partie 1

TP Filtrage

28 novembre 2014

1 Addition de Minkowski :

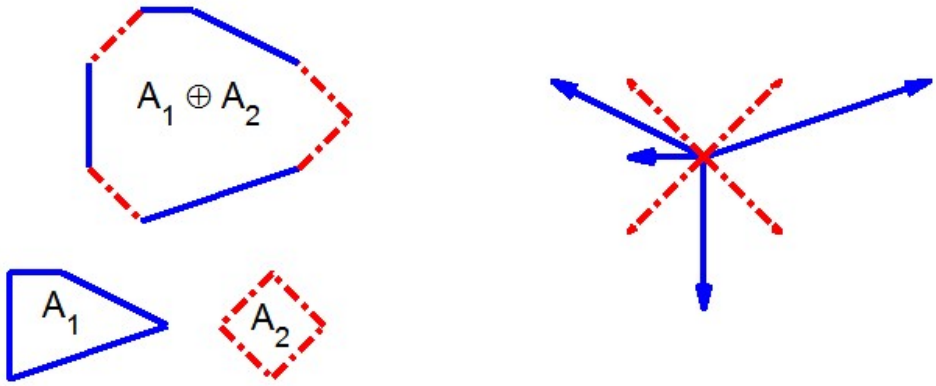

FIGURE 1 – *Addition de Minkowski de deux polygones par interclassement des vecteurs d'arêtes.*

Etant donné deux "formes" *convexes* A et B, la somme de Minkowski de ces deux formes, notée $C = A \oplus B$, est l'ensemble :

$$C = \{\vec{w} = \vec{u} + \vec{v} \text{ avec } \vec{u} \in A \text{ et } \vec{v} \in B\}$$

Propriétés de l'opérateur $\oplus$:
a) commutativité : $A \oplus B = B \oplus A$.
b) associativité : $(A \oplus B) \oplus C = A \oplus (B \oplus C)$.
c) existence d'un neutre : $A \oplus \emptyset = A$.

Lorsque les formes en question ont des contours polygonaux, il en est de même de leur somme de Minkowski. On montre que ce contour peut être obtenu en procédant à un interclassement des vecteurs $\overrightarrow{V_k} =$

Plus précisément, à un polygone défini par ses n sommets $M_1, ...M_n$ dans le plan "à translation près", on peut associer le "bouquet de n vecteurs" définis ci-dessus (convention $M_{n+1} = M_1$). Cette association fonctionne aussi (bijectivement) dans l'autre sens : le cumul progressif des $\overrightarrow{V_k}$ redonne les M_k. Si l'on note B cette association de "prise en bouquet", on a donc l'identité

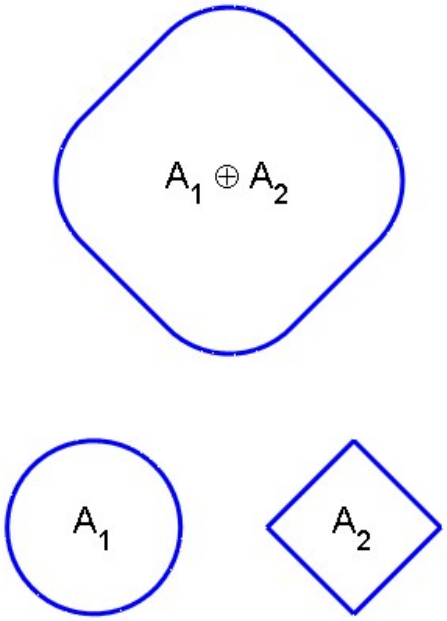

FIGURE 2 – *L'addition de Minkowski d'un polygone et d'un disque de rayon r est le dilaté du polygone de taille r.*

$B(P_1 \oplus P_2) = B(P_1) \cup B(P_2)$. Si bien que l'on peut résumer l'algorithme à ceci :

En entrée P_1 et P_2 (décrits par leurs sommets).
$Bouquet = B(P_1) \cup B(P_2)$.
$P_3 = B^{-1}(Bouquet)$ est la somme de Minkovski de P_1 et P_2.

Cette façon de "redéfinir" l'opérateur $\oplus$ permet notamment de prouver sa commutativité et son associativité de façon quasi immédiate.

Nous allons travailler dans le plan complexe, car il s'avère que les fonctions que l'on peut utiliser y sont globalement plus simples.

Vous trouverez ci-dessous un programme qu'il s'agit dans un premier temps de comprendre, en le commentant abondamment, puis, dans un deuxième temps, de compléter au niveau des lignes centrales (utiliser l'opérateur 'diff' et les fonction 'abs' et 'angle') .

```
close all;clear all;hold on;axis equal
i=complex(0,1);
P1=[0;1;1+i;i;0];
P2=[2;4;3+i;2];
...
...
...
...
```

```
...
[An,Ind]=sort(A);
S=R(Ind).*exp(i*An);
P3=cumsum(S);P3=[P3;P3(1)];
P=[P1;NaN;P2;NaN;(4.5+i*4.5)+P3];
plot(real(P),imag(P),'LineWidth',2,'linesmoothing','on');
text([0.4,2.9,4.6],[0.5,0.3,3.5],...
  {'A_1','A_2','A_1 \oplus A_2'},'fontsize',14);
```

On pourra (voir Fig. 2) prendre pour P_1 le polygone régulier des racines n-ièmes de l'unité pour réaliser une dilatation approchée de P_2 par un disque.

2 Addition de Minkowski par convolution de 2 images :

Que fait le programme suivant ? Quel rapport voyez-vous avec la convolution de Minkowski ?

```
clear all;close all;
n=256;
I=zeros(n);
for i=1:n
    for j=1:n;
        I(i,j)=((i+j)>(n-20))&&((i+j)<(n+20))&&(i>110)&&(i-j<130);
    end;
end;
figure(1)
imshow(I);
J=zeros(n);
for i=1:n
    for j=1:n;
        J(i,j)=(i-n/2)^2+(j-n/2)^2<=400;
    end;
end;
figure(2)
imshow(J);
K=conv2(im2double(I),im2double(J),'same');
K(100:160,100:160);% permet de se rendre compte des valeurs des pixels
figure(3);
imshow(K>0);
%imagesc(K);% à décommenter
S=strel('disk',20);
L=imdilate(I,S);
imshow(L)
L(100:160,100:160)
figure(4)
imshow(abs((K>0)-L));
```

3 Filtrage moyenneur :

Commentez le programme suivant ; comparer le résultat à celui d'un filtre gaussien w=fspecial('gaussian',100,sigma), avec par exemple sigma = 3.

```
clear all;close all;
I=imread('barbara.png');% on pourra prendre également I=checkerboard(100,5,5);
whos I;%  dans le cas de ''barbara'', c'est une image 512 x 512 de type uint8
figure(1);imshow(I)
n=10;
w=(1/(2*n+1)^2)*ones(2*n+1);
%filtre moyenneur 21 x 21 ; notez les bords; essayez d'autres valeurs de n
figure(2);J=imfilter(I,w,'same');imshow(J);
figure(3);J=imfilter(I,w,'replicate');imshow(J);%évite les bords
```

4 Filtrage dérivateur :

On a vu dans le TP précédent qu'un filtre dérivateur selon l'axe Ox est donné par la matrice :

$$\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$$

En faisant de même avec sa transposée, on peut en déduire l'image de la norme du gradient $\sqrt{Gx.^2 + Gy.^2}$, à moins que l'on ne préfère, plus simplement, $abs(Gx) + abs(Gy)$.

a) Ecrire un programme qui compare, sur l'image "Barbara", la norme du gradient avec le filtre de Sobel.

Indications : utiliser :
Gx=im2double(imfilter(I,m,'same')) ;
imfilter(I,fspecial('sobel'),'replicate') ;

b) Comment pourrait-on imaginer un filtre "dérivée seconde" ?

Solution :

a) Voici les instructions manquantes au premier programme :

```
V1=diff(P1);
V2=diff(P2);
Vf=[V1;V2];%fusion
A=angle(Vf);
R=abs(Vf);
```

Comparaison entre filtres dérivateurs ($\sqrt{(}Gx^2 + Gy^2)$) et filtre de Sobel :

```
clear all;close all;
I=imread('barbara.png');
m=[-1,0,1;
   -2,0,2;
   -1,0,1];
mr=rot90(m);
Ip=im2double(imfilter(I,m,'same'));
Iq=im2double(imfilter(I,mr,'same'));
figure(1);J=sqrt(abs(Ip).^2+abs(Iq).^2);imshow(J);
Isobel=imfilter(I,fspecial('sobel'),'replicate');
figure(4);imshow(Isobel)
```

Ils ont même allure...

Partie 2

2 Déconvolution

2.1 Qu'est-ce que la déconvolution

La déconvolution est le fait de defaire une operation de convolution. Par exemple, lorsqu'on fait une mesure avec un appareil, cet appareil peut se modeliser par sa reponse impulsionnelle (reponse a un dirac) qui caracterise comment l'appareil deforme la grandeur mesuree. De plus, il ne faut pas oublier qu'il peut y avoir du bruit qui vient s'ajouter a notre mesure. Plus formellement, si on definit i la mesure que l'on souhaite avoir, h la reponse impulsionnelle, η du bruit de mesure et y la mesure observee, on peut ecrire le probleme direct comme :

$$y = i * h + \eta$$

ou $*$ represente le produit de convolution. La deconvolution consiste a retrouver i... ce qui n'est pas forcement une operation facile surtout dans le domaine spatiale. Une approche plus simple est de voir le probleme dans le domaine frequentiel, ce qui s'ecrit :

$$Y = I * H + N$$

avec Y, I, H et N les transformees de Fourier de y, i, h et η respectivement. On vous propose de tester trois methodes de deconvolution (vous verrez que deux sont vraiment similaires). On commence par un filtrage inverse, puis filtrage inverse avec seuil et enfin un methode un peu plus avancee, le filtre de Wiener.

Simulation de l'aquisition d'une image Pour simuler une acquisition, nous vous proposons de d'utiliser une reponse impulsionnelle gaussienne, ce qui dans beaucoup de cas est une bonne approximation (ex : appareil photo).

$$h\left(x,y\right) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

avec sigma un parametre de la modelisation. Pour la simulation, proceder a la premiere etape :

- chargez (*imread*) l'image de Lena et visualisez la (ou une autre image en niveau de gris... il faut toujours verifier la dimension de l'image en profondeur)
- calculez la transformee de Fourier de Lena et affichez la a cote de Lena dans l'espace direct
- creez le model de l'appareil. Utiliser la fonction *meshgrid* pour generer les matrices X et Y. N'oubliez pas de fixer σ. Visualisez le modele
- prenez la transformee de Fourier du modele et visualisez sa valeur absolue et sa phase. Utilisez les fonction *fft2* et *fftshift* pour calculer et re-arranger la transformee de Fourier
- multipliez, dans le domaine de Fourier, l'image de Lena et le modele.
- revenez dans le domaine direct et observez l'influence de σ. Servez vous de *ifft2* et *ifftshift*
- pour finir la simulation, ajouter du bruit gaussien centre dont vous fixerez la variance σ_{b}^2

Filtrage inverse

On suppose que l'on connait le modele de notre appareil (σ), et on cherche a faire la deconvolution sans prendre de precotion.

- calculez le filtre inverse, qui est literalement l'inverse point a point du modele H (dans le domaine de fourier)
- multipliez dans le domaine de Fourier l'image et le filtre inverse.
- visualisez la transformee de Fourier de l'image d'origine (I) et le resultat precedent
- revenez dans le domaine direct, et comparez votre resultat a l'image d'origine et celle acquise.
- testez avec differents parametre du modele et niveau de bruit.

Filtrage inverse avec seuillage

Ce filtrage est la suite logique du filtrage inverse. Il suffit de refaire les etapes precedentes en seuillant le filtre inverse pour ses valeurs trop faible qui peuvent induire une instabilitee lors de l'inversion, par exemple a cause du bruit. Essayez de gerer les situation ou le filtre inverse est trop faible... comment faire?

Filtre de Wiener

Ce filtre est un filtre qui cherche a repondre a la question : comment faire pour adapter la force du filtre en fonction du niveau de bruit. Pour le definir, on cherche la fonction G telle que

$$\hat{i} = g * y, \qquad g = \arg\min_{g} \quad \mathbb{E}\left[\left(i - \hat{i}\right)^2\right]$$

avec G la transformee de Fourier de g et son expression est

$$G = \frac{\overline{H}S}{|H|^2 S + N} = \frac{1}{H}\frac{|H|^2}{|H|^2 + \frac{1}{\mathrm{SNR}}}$$

ou $\overline{H}$ est le conjugue complexe de H, et S est la densitee spectrale de puissance definie par :

$$S_{i,i}\left(f_x, f_y\right) = \mathcal{F}\left\{\left(i * i\right)\left(-\tau_x, -\tau_y\right)\right\}$$

et le SNR est le rapport signal a bruit definit comme le rapport des puissances du signal sur celle du bruit

$$\mathrm{SNR} = \frac{P_{\mathrm{signal}}}{P_{\mathrm{bruit}}}$$

avec la puissance definit comme :

$$P = \lim_{x_1 \to \infty} \lim_{x_2 \to \infty} \frac{1}{2x_1}\frac{1}{2x_2}\int_{-x_1}^{x_1}\int_{-x_2}^{x_2} |i|^2 \,\mathrm{d}x_1 \mathrm{d}x_2$$

pour tester ce filtre, suivez les etapes :

- estimez le SNR de l'image. Rappel : la puissance d'un bruit gaussien est sa variance
- calculer la deuxieme formule du filtre
- appliquez le filtre a l'image
- observez le resultats

On se propose de regarder la premiere formulation du filtre

- estimez la densite spectrale de puissance de l'image en utilisant les fonctions *corr2* et *fft2*
- calculer la premiere formule du filtre
- appliquez le filtre a l'image
- observez le resultats

3 Segmentation

La segmentation consiste a associer une classe d'appartenance a des echantillon $x(i,j)$. Dans notre cas, l'echantillon est l'image et la classe reperee par un label c_k est caracterisee par sa moyenne μ_k pour $k \in [\![1, K]\!]$. On cherche a segmenter l'image au sens de du minimum de la valeur :

$$T = \sum_{k \in [\![1,K]\!]} \sum_{\text{pixel}(i,j)} (x(i,j) - \mu_k)^2$$

l'algorithme qui minimise cette valeur est l'algorithme des k-moyennes. Qui fonctionne comme indique ci apres :

0 Initialisation des centre μ_k des classe c_k

1 affectation de chaque pixel a sa classe la plus proche

2 calcul des nouvelles valeurs μ_k

3 refaire les etape 1 et 2 jusqu'a convergence

Implementez une fonction qui realise cet algoritme et l'utilise sur l'image de Lena.