TP Morphologie mathématiques : les opérateurs élémentaires et leurs applications

November 21, 2014

1 introduction

Rappel 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 Objectif

Ce premier TP du module de TSI vous permet de vous familliariser avec :

- la manipulation des données images via les fonctions de chargement comme *imread*, l'affichage *imshow* ou histogramme comme *hist* et bien d'autres.
- les opérateurs elementaire de morphologie mathematiques
- des pplications un peu plus avancees se basant sur les operateurs d'erosion, dilatation, ouverture et fermeture.

1.2 À rendre

De votre part, nous attendons deux elements :

- Un compte rendu explicant votre demarche, 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 necessaire (toujours commentees)

1.3 Les moyens

Vous trouverez toutes les resources (resume de cours, enonce de TP et des morceaux de script à completer) à l'adresse suivante : ...

Consigne à observer Le TP doit se derouler sous OCTAVE que vous pouvez lancer soit en mode terminal via la commande *octave* soit en demarrant 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 qualite (contrairement au capture d'ecran). Pour plus d'information sur cette commande, nous vous invitons a taper dans votre prompt octave *help print*.

2 Prise en main

Dans cette section vous allez voir comment faire pour acceder aux donnees d'une image et la transformee. Commencez par creer un fichier "TP_moprho_init.m" qui commencera par les trois lignes suivantes :

clc

clear all

close all

pour eviter d'avoir des conflit de definition et des affichages multiple en navigant entre les fichiers.

2.1 Affichage d'une image

Testez le script suivant et decrivez ce qu'il fait. Approtez des modifications si necessaire.

```
clear all;
close all;
clc;

%chargement de l'image dans un tableau
imRGB = imread('accessoire_poker_heart.jpg');

%affichage de l'image
figure(1)
imshow(imRGB)
title('mon titre')

%autre methode d'affichage
figure(2)
imagesc(imRGB)
title('mon autre titre')
%colormap jet %ligne a decommenter pour tester
%colorbar

%transformation de l'image en niveau de gris methode 1
class(imRGB(1)) %affiche le type de variable
imGray1 = imRGB(:,:,1) + imRGB(:,:,2) + imRGB(:,:,3); %somme des trois composantes... mais est-ce

%transformation de l'image en niveau de gris methode 2
imGray2 = rgb2gray(imRGB);

figure(3)
subplot(121)
imshow(imGray1)
title('methode de transformation : somme des composantes')
subplot(122)
imshow(imGray2)
title('methode de transformation : commande rgb2gray')
```

2.2 Binarisation

Maintenant que vous savez charger une image en memoire et que vous pouvez la transformee en niveaux de gris, vous allez chercher a binariser une image et a extraire des contours. Vous utiliserez le script suivant en apportant les modifications necessaires et vous expliquerez ce que fait le script.

```
clear all;
close all;
clc;

%chargement de l'image dans un tableau
imRGB = imread('accessoire_poker_heart.jpg'); %faut il changer de type...

%transformation de l'image en niveau de gris
imGray = ...;

%detection des contours par seuillage de la norme du gradient (estimation par filtre de sobel)
    %definition des filtre pour l'estimation du gradient
Fl = (1/4)*[-1 -2 -1;
         0   0   0;
         1   2   1];
Fc = Fl';
    %estimation des composantes du gradient
imGradl = conv2(imGray, Fl,'same');
imGradc = conv2(imGray, Fc,'same');

    %calcul de la norme du gradient
imGradNorm = ...;

    %seuillage de la norme du gradient pour extraction des contours
        %pour savoir a quelle valeur seuiller, on affiche l'histogramme de l'image a l'aide de la
imhist...
        %pour fixer le seuil, on choisit de demander a l'utilisateur le taper dans le prompt
seuil = input("nom text\n")
        %seuillage de la norme du gradient (pas de boucle autorisee)
imGradBin = ...;

%affichage des resultats
figure(1)
...
```

3 Observation des operaeurs de base

3.1 Somme et soustraction de Minkowski

En vous aidant des notes de cours (mopho_resume_cours.pdf) et des fichier minkowskiSum.m et minkowskiSub.m, vous implementerez les operations d'addition et de soustraction Minkowskiennes. Attention, votre code ne doit pas contenir plus de deux boucles **for**. Vous testerez vos implementations sur des elements simple comme un cercle avec un carre.

Attention : ces deux codes sont tres importants car toutes les autres parties reposent dessus, il faut donc que vous pretiez une attention toute particuliere a la validation de cette etape.

Conseil : pour la validation, vous pouvez creer des objet simple comme le disque centre sur l'origine et de rayon 5 que realise le script suivant.

```
[dC,dL] = meshgrid(-49:50,-49:50);
X = sqrt(dC.^2 + dL.^2)<5;
```

A rendre : explication du script et de sa validation.

3.2 Erosion, dilatation, ouverture et fermeture

Implementation Implementez les operateurs d'erosion et de dilatation a l'aide des somme et soustraction de Minkowski. Implementez ensuite l'ouverture et la fermeture en utilisant les operateur que vous venez de coder.

Validation Dans une premier temps, tester vos algorithmes sur les images qui vous sont fournis (certaine sont binaire comme "veau2_moins_petit.bmp" et peuvent etre utilisees sans pretraitement). Vous pouvez aussi comparer vos resultats a ceux des fonction *imerode*, *imdilate*, *imopen* et *imclose*.

A rendre : Test de plusieurs elements structurants sur une image de votre choix dans le jeu d'image fournis. Mise en evidence des proprietes de composition et de dualite pour erosion/dilatation et propriete d'idempotence et de dualite pour ouverture/fermeture.

3.3 Hit or Miss

Implementation En utilisant les operateurs precedemment codes, implementer la transformee Tout-ou-Rien.

Validation vous testerez sur l'image "bin.png" en ajoutant du bruit (des points aleatoirement distribues pour lesquels l'etat du pixel est complemente) puis en la debruitant. Indication : utilisez

```
N = randn(300,400)>2.0;
```

pour generer le bruit et la commande *xor* pour ajouter le bruit a l'image. Vous pourrez comparer votre resultat a celui de **bwhitmiss**.

A rendre : la preuve en image que votre implementation fonctionne avec une explication de la methode que vous utilisez pour debruiter l'image.

4 Applications

4.1 Detection de periode

4.2 Granulométrie

4.3 Comptage de trou

4.4 Gradient... comparaison avec Hit-or-Miss

4.5 Squelette

3) Reconstruction par marqueur.

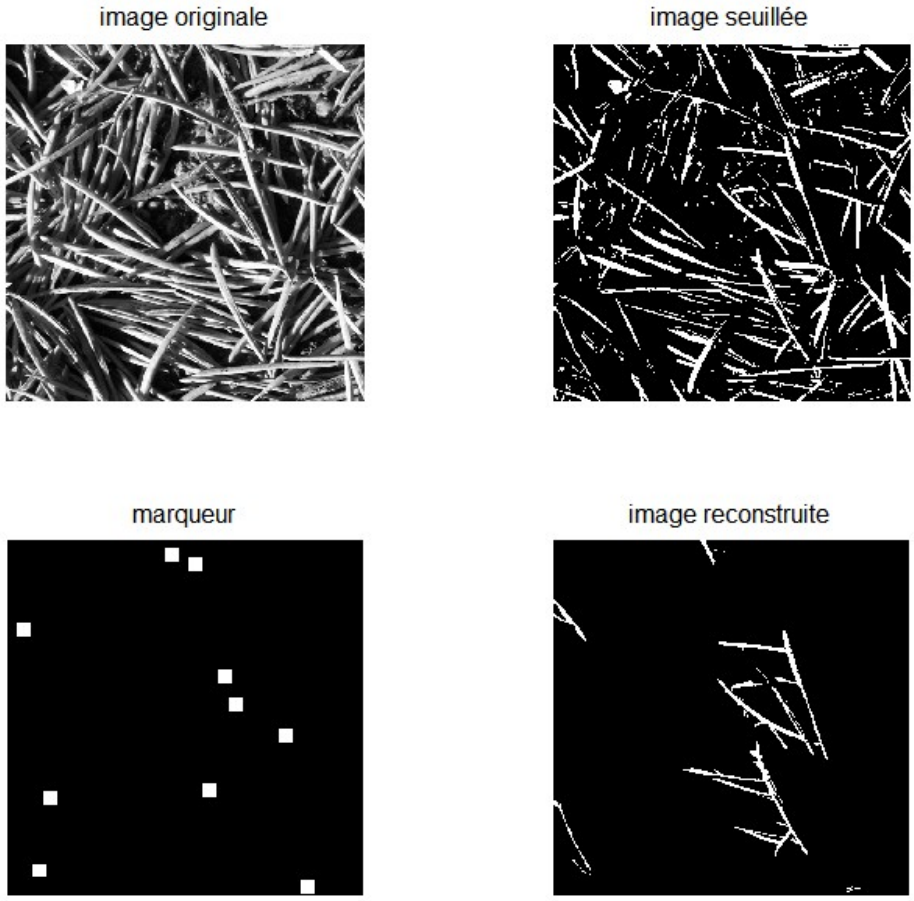

FIGURE 1 – *Image d'un tapis d'épines de pin, seuillée pour en faire une image binaire. On construit une image "marqueur" constituée de 10 "points" indiquant des zones à reconstruire. On reconstruit une nouvelle image constituée des zones blanches de l'image seuillée qui avaient au moins un pixel commun avec un pixel blanc de l'image marqueur.*

Pour comprendre la reconstruction par marqueurs, tapez le programme Matlab ci-dessous dont une trace d'exécution est donnée par la Fig. 1. Comprenez-en les différentes étapes et commentez-les.

```
clear all;close all;clc;
set(gcf,'color','w');%fond blanc
f=imread('aiguilles_pin.jpg');%à remplacer par une image de votre choix
f=f(1:512,1:512);
[a,b]=size(f);
subplot(2,2,1);imshow(f);title('image originale');
f=(f>200);
whos f
f=im2double(f);
subplot(2,2,2);imshow(f);title('image seuillée');
marq=zeros(a,b);% équivalent à : m=zeros(size(f));
for k=1:10;
    marq(ceil(a*rand),ceil(b*rand))=1;
end;
marq=imdilate(marq,ones(20),'same');
subplot(2,2,3);imshow(marq);title('marqueur');
g=imreconstruct(marq,f);
subplot(2,2,4);imshow(g);title('image reconstruite');
```

Travail personnel : Comment pourrait-on se servir de la reconstruction par marqueur pour éliminer les aiguilles qui touchent le bord de l'image (indication : prendre comme marqueur le bord de l'image)