Traitement et analyse d'images

Travaux pratiques

TP N°2 : Segmentation – Morphologie Mathématique

---

1. Contexte

Un des problèmes essentiels dans l'analyse d'une image est l'extraction des différentes entités, appelées objets ou régions, qui la constituent. Ces entités doivent être représentatives, bien correspondre à l'information désirée et plus aisément traitables que l'image toute entière. Tout cela relève de la segmentation. Cette étape conditionne tous les traitements ultérieurs car une mauvaise partition initiale entraîne inévitablement des erreurs dans les mesures, les classifications, les reconnaissances, les interprétations, etc.
Il n'existe pas de méthode universelle de segmentation mais seulement des méthodes adaptées à certains types d'images et à certains objets recherchés.

La Morphologie mathématique utilise des opérations ensemblistes pour simplifier une image après transformation. Il y a perte d'information, mais les caractéristiques essentielles sont mises en évidence. En terme de prétraitement et/ou de segmentation :

- une érosion supprime les petites particules indésirables, réduit la taille des autres particules, supprime les pics et peut déconnecter certaines particules.
- une dilatation bouche les petits trous à l'intérieur des particules et les golfes des frontières, augmente la taille des particules et peut connecter certaines particules proches.
- une ouverture supprime les petites particules, élimine les détails fins du contour en lissant par l'intérieur; elle peut déconnecter certaines particules.
- une fermeture bouche les trous à l'intérieur des particules, élimine les détails fins du contour en lissant par l'extérieur; elle peut connecter des particules proches.

2. Détection de Contours

Intuitivement, un contour peut être associé à une variation d'intensité ou à une discontinuité entre les propriétés de deux ensembles connexes de points. Pour détecter et localiser ces variations ou discontinuités, les approches dérivatives sont les plus immédiates et les plus souvent utilisées. Les contours sont assimilés aux points de fort gradient ou de dérivée seconde nulle.

Illustrons maintenant ces approches par les deux séquences suivantes :

Séquence 1 :

```
load trees
ImB = ind2gray(X,map);
h     = fspecial('sobel')  // ôter le ';' afin d'afficher la matrice du filtre
ImF = filter2(h,ImB);
colormap(gray(256)), imagesc(ImF), axis off
```

Pourquoi convertit-on l'image en niveaux de gris ? Pourquoi utilise-t-on la fonction **imagesc** plutôt que **imshow** ? Appliquer et commenter les effets du filtre '**Sobel'** en horizontal et vertical.

Séquence 2 :

```
I=imread('objects.tif');
figure(1), imshow(I)
figure(2), imshow(~edge(I)) // 'sobel' par défaut
figure(3), imshow(~edge(I,'roberts'))
figure(4), imshow(~edge(I,'prewitt'))
figure(5), imshow(~edge(I,'log'))  // le paramètre échelle σ = 2 par défaut
figure(6), imshow(~edge(I,'canny')) // le paramètre échelle σ = 1 par défaut
```

Remarques et conclusions ?

Séquence 3 :

```
close all
load trees
I = ind2gray(X,map);
figure(1), imshow(I)
figure(2), imshow(~edge(I)) // 'sobel' par défaut
figure(3), imshow(~edge(I,'roberts'))
figure(4), imshow(~edge(I,'prewitt'))
figure(5), imshow(~edge(I,'log')) // le paramètre échelle σ = 2 par défaut
figure(6), imshow(~edge(I,'canny')) // le paramètre échelle σ = 1 par défaut
```

Remarques et conclusions ?

Séquence 4 :

```
close all
load trees
I = ind2gray(X,map);
figure(1), imshow(I)
[BW,S]=edge(I,'canny');
S
figure(2), imshow(~edge(I,'canny'))
figure(3), imshow(~edge(I,'canny',S/2))
figure(4), imshow(~edge(I,'canny',2*S))
```

Remarques et conclusions ?

Séquence 5 :

```
close all
load trees
I = ind2gray(X,map);
[BW,S]=edge(I,'canny');
S
figure(1), imshow(~edge(I,'canny'))  // le paramètre échelle σ = 1 par défaut
figure(2), imshow(~edge(I,'canny',S,2))
figure(3), imshow(~edge(I,'canny',S,3))
figure(5), imshow(~edge(I,'canny',S,5))
figure(7), imshow(~edge(I,'canny',S,7))  // le paramètre échelle σ = 7
```

Remarques et conclusions ?

```
close all
[BW,S]=edge(I,'log');
S
figure(1), imshow(~edge(I,'log',S,1))
figure(2), imshow(~edge(I,'log',S,2))
figure(3), imshow(~edge(I,'log',S,3))
figure(4), imshow(~edge(I,'log',S,4))
```

Remarques et conclusions ?

Séquence 6 :

```
close all
Obj = imread('objects.tif');
Cel = imread('cell.tif');
figure(1), imshow(Obj)
figure(2), imshow(Cel)
h     = fspecial('laplacian')  // ôter le ';' afin d'afficher la matrice du filtre

LAP = filter2(h,Obj);
ZC1 = (LAP == 0.0);
ZC2 = (LAP<=5 & LAP>=-5);
figure(3), imshow(~ZC1)
figure(4), imshow(~ZC2)
figure(5), imshow(~edge(Obj,'log'))

LAP = filter2(h,Cel);
ZC1 = (LAP==0.0);
ZC2 = (LAP<=5 & LAP>=-5);
figure(3), imshow(~ZC1)
figure(4), imshow(~ZC2)
figure(5), imshow(~edge(Cel,'log'))
```

Remarques et conclusions ?

3. Seuillage et Segmentation

- ***Segmentation et Squelettisation***

Effectuer et commenter la séquence suivante :

```
ImObj = imread('objects.tif');
figure(1), imshow(ImObj)
level = graythresh(ImObj)
BW = im2bw(ImObj, level);
figure(2), imshow(BW)
SQ = bwmorph(BW,'skel',20);
figure(3), imshow(SQ)
Result = xor(BW,SQ);
figure(4), imshow(Result)
```

Conclusions.

4. Morphologie Binaire

▪ *Érosion et Dilatation*

Matlab permet de définir l'élément structurant et sa taille, en fonction du traitement que l'on veut réaliser.
Rechercher l'aide sur les transformations morphologiques de base **imerode** et **imdilate**. Effectuer et commenter ensuite les instructions de la séquence suivante :

```
BW = imread('morpho0.tif');
figure(1), imshow(BW)
ER = imerode(BW,strel('disk',4));
DL = imdilate(BW,strel('disk',4));
figure(2), imshow(ER)
figure(3), imshow(DL)
```

Sur la même image, effectuer et comparer des érosions et dilatations de différentes tailles (2, 3, 4, ...). Conclusions.

▪ *Ouverture et Fermeture*

À partir de l'image binaire **morpho1.tif**, effectuer une ouverture de taille 4, une fermeture de taille 4 et une fermeture d'ouverture de même taille :

```
BW  = ...
OP4 = ...
CL4 = ...
CL_OP4 = ...
```

Commenter et Interpréter les images résultantes. Conclusions.

5. Quelques Applications

- ***Extraction et Caractérisation***

Lire et afficher l'image en niveaux de gris **blockn.tif**. Proposer et effectuer une suite de traitements permettant d'extraire l'objet de forme carrée situé dans la partie supérieure droite de l'image (sans utiliser la commande **roipoly**). Mesurer ensuite l'aire totale du carré (surface du disque central incluse) en utilisant (**'bwarea'**), et le périmètre du carré en utilisant **'bwperim'**. Les opérations effectuées devront respecter au maximum les dimensions initiales du carré.
Conseil : Vous pouvez commencer par analyser l'histogramme de l'image, puis penser à une segmentation par seuillage.

- ***Reconnaissance et comptage***

Lire et afficher l'image **pillsetc.png** (image RGB. Proposer et effectuer une suite de traitements permettant de déterminer le nombre d'objets de forme circulaire (ou quasi-circulaire).

6. Segmentation d'image texturée

Traditionnellement, la représentation de contours est déterminée automatiquement comme étant à la fois une opération de détection et de chaînage ou de suivi de contours. Cette approche classique souvent adoptée, présente de nombreuses limitations dans le cas d'images bruitées et texturées. En effet, les opérateurs locaux utilisés font preuve d'une forte sensibilité vis-à-vis de la texture des objets traités et du bruit dû aux mauvaises performances du système d'acquisition. Précisément, la principale difficulté rencontrée en détection de contours réside dans le choix des seuils, qui sont le plus souvent déterminés de manière empirique par un utilisateur, à la suite d'essais répétés. L'inhomogénéité d'acquisition et l'absence de seuil unique pour ces images texturées font que cette méthode de détection de contours se voit placée devant un problème d'identification soit de peu soit de trop de points de contour. Ces opérateurs locaux ne prennent pas en compte l'information globale sur le contour, information pouvant être issue d'un modèle a priori de contour. De plus, ils nécessitent une phase de post-traitement afin d'assurer la fermeture des contours. Cette dernière opération souffre d'un manque de perception globale sensible et organisée de l'information locale trouvée. Afin de pallier ces problèmes et d'améliorer par la suite les résultats dans ce type d'images, il apparaît donc nécessaire d'introduire explicitement, dès la phase de détection, des informations sur les caractéristiques géométriques des contours, principalement sur leur continuité et leur régularité. D'ailleurs, le schéma de reconnaissance de formes utilisé par un observateur humain est un exercice de perception visuel dans lequel la notion de continuité de structures joue un rôle primordial. Ces techniques de détection de contours sont regroupées sous le nom de "approches par modèles déformables".

Etudier et effectuer différents seuillages et segmentations de l'image **otolithe.tif**. Interpréter et Commenter les différentes images résultat. Conclusions ?
