Traitement d'images : TP4

Terminer les exercices du TP précédent.

Exercice 1 Reconstruction géodésique On considère l'image binaire présentée en TD représentée par la matrice I0 suivante.

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0	0	1	0	0
0	0	1	1	1	0	0	0	0	1	0	0	0	0	0
0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	0	0	0
0	1	0	1	1	1	1	1	1	1	0	1	0	0	0
0	0	0	1	1	1	1	1	1	1	0	1	0	0	0
0	0	0	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	1	0	0	0	0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0	0	1	0	0	1	0	0
0	0	0	1	0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

On prend comme élément structurant un carré centré de taille 3. On effectue une reconstruction géodésique. Cette opération se définit de la manière suivante :
I0 ← Image originale
  I1 ← ouverture(I0)
  Faire
    I2 ← dilatation(I1)
    I3 ← minimum pixel à pixel entre I2 et I0
    test ← I3-I1
    I1 ← I3
  Tant que test ≠ image nulle

Implémentez l'algorithme et testez sur l'image I0. Affichez les résultats intermédiaires.
Remarque : pour tester si la matrice test est nulle, on peut utiliser `any(test(:))`.

Exercice 2 Segmentation
On veut écrire un programme de segmentation basée sur les valeurs des pixels.

1. Seuillage multiple
   Ecrire les fonctions effectuant un seuillage multiple à partir d'un nombre de classes donné. Tester les fonctions sur les images `Image1.bmp` et `Image2.bmp` en faisant varier le nombre de classes et enregistrer les images résultats.

2. Affichage en fausses couleurs
   Si on affiche le résultat du seuillage (ou de la segmentation) à partir de la matrice résultat, l'image ne sera pas très lisible. Pour bien distinguer les classes entre elles, on associe des couleurs aux classes (couleurs qui n'ont rien à voir avec les couleurs originales de l'image, d'où le terme "fausses couleurs"). Pour cela, on définit une palette de couleurs et on affiche l'image résultat comme s'il s'agissait d'une image indexée.
   Ecrire la fonction qui crée une palette de 20 couleurs différentes (choisir des couleurs bien distinctes). Afficher ensuite le résultat des seuillages précédents en fausses couleurs.

Exercice 3 Etude des paramètres des régions
Créer une version binaire de l'image `coins.png` par seuillage. La fonction `im2bw` permet de convertir l'image en niveaux de gris en une image binaire par seuillage. La fonction `graythresh` calcule automatiquement le seuil approprié pour la conversion.

```
level = graythresh(I);
bw = im2bw(I,level);
```

Pour combler les trous de l'image, utiliser la fonction `imfill` de matlab.

Pour déterminer le nombre de pièces présentes sur l'image, vous pouvez utiliser la fonction `bwlabel` en donnant le nombre de composantes connexes souhaitées.

La fonction `regionprops` permet de mesurer les propriétés des régions d'une image. Regarder l'aide Matlab et afficher pour chaque pièce :

— les coordonnés du centre
— le diamètre
— l'aire

Afficher sur un fond noir

— les 4 plus petites pièces
— les 3 plus grandes pièces
— les 2 plus claires
— la plus foncée

Pour trier un vecteur, on peut utiliser la fonction `sort` de matlab.