Mathematical Morphology - Watershed segmentation

Assignment 6

`Nom :`
`Prénom :`

In this assignment, you will learn about the watershed transformation and its applications: separation of overlapping blobs and segmentation. You will use the Matlab function 'watershed' which can find the catchment basins and watershed lines for any grayscale image.

Contents

PART 1 - Distance transform and Watershed

Combined with a distance transform, the watershed transformation can help separating touching objects. In this part, you will study the 2D example provided by Matlab. This example presents the task of separating two touching objects in a binary image. Second, you will apply the method to separate the touching circles in a binary image.

Links :

— http://. . . /the-watershed-transform-strategies-for-image-segmentation.html
— http://cmm.ensmp.fr/~beucher/wtshed.html

Explications du code de l'exemple de matlab

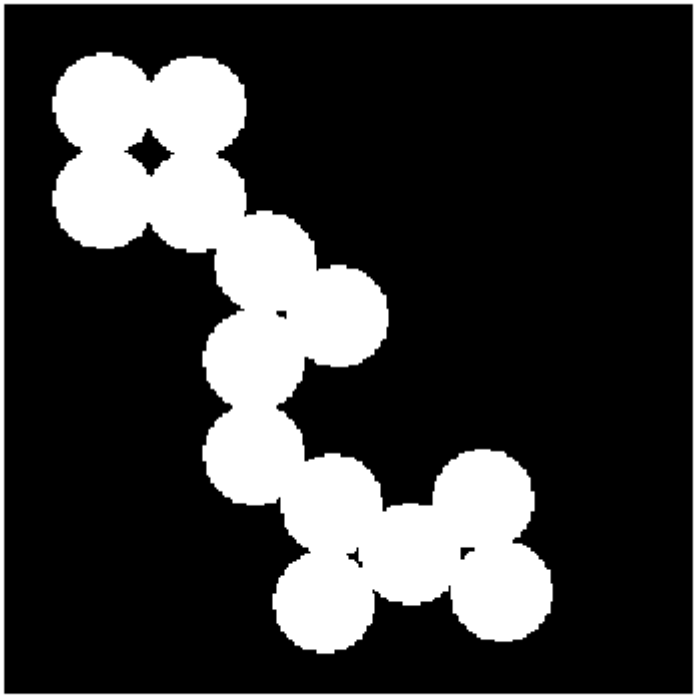
a) In step b), why is the distance transform applied to the complement of the binary image, not to the image ?
   Réponse : sinon on calculerait la fonction distance en dehors des objets
b) In step c), why do we complement the image D ?
   Réponse : pour que les max de la fonction distance deviennent des min
c) In step c), explain the instruction D(~bw) = -Inf
   Réponse :

Le signe ~ correspond à l'opérateur NON logique. Lorsque les valeurs sont égales à 0, elles prennent la valeur 1 et sinon, elles prennent la valeur 0. Le résultat est une image binaire (logical).
D(~bw) = -Inf : permet de mettre la valeur -Inf au niveau des pixels du fond égaux à 0 afin de créer un minimum au niveau du fond et obtenir les frontières des objets.

1.1. Segmentation of connected circles

The studied image is a binary image of connected circles.

```
clear all;
close all;

IBW=imread('circles.png');
figure(1), imshow(IBW)
```



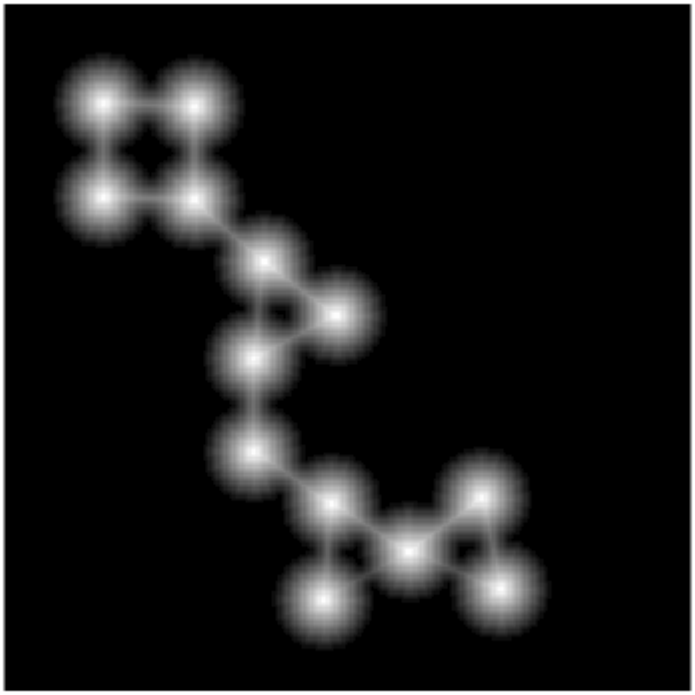
1.2. Segment (and disconnect) the circles using a distance transform

Pour calculer la distance intra-objets, il faut utiliser le complementaire de l'image.

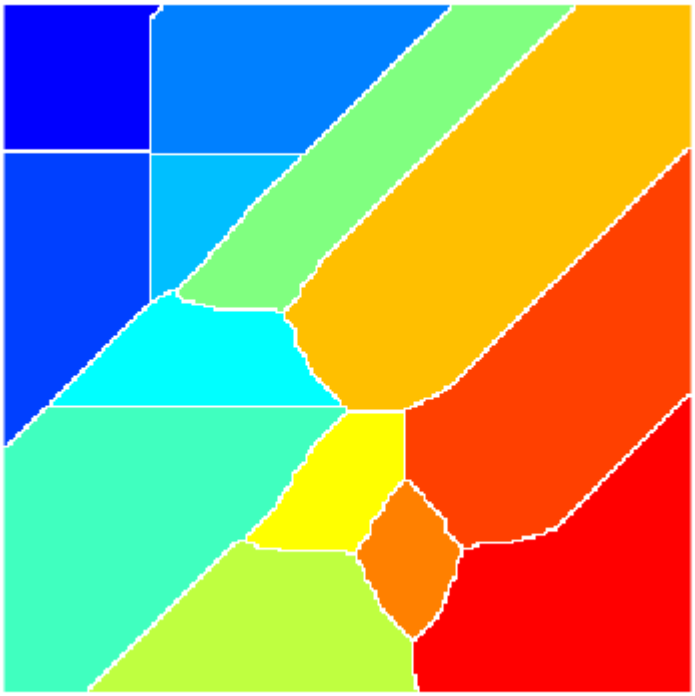
```
D=bwdist(~IBW);

figure(2), imshow(D,[ ]), title('fonction distance')
```

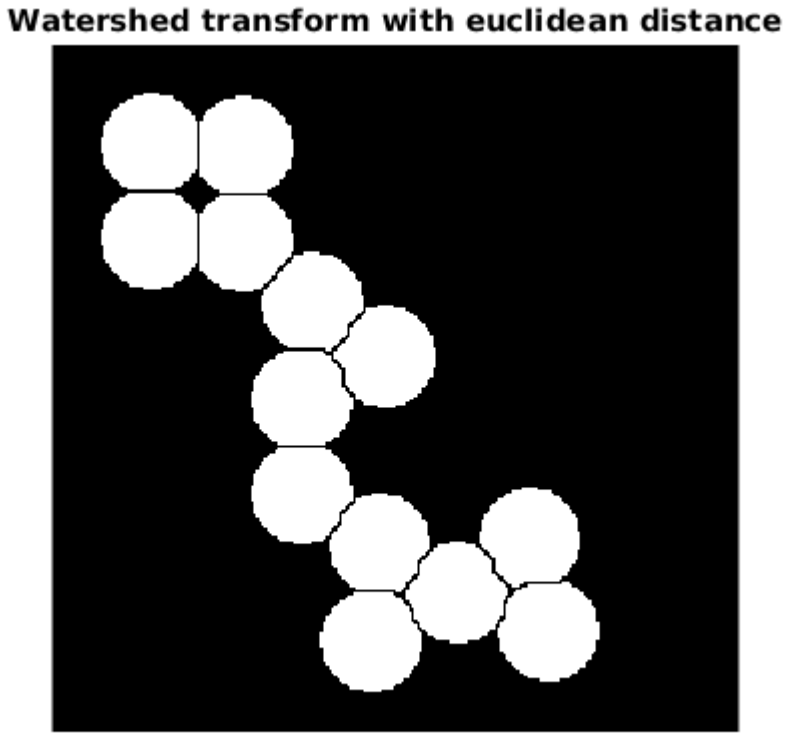
```
% Le fond est à 0, dans les objets, la distance augmente lorsque l'on
% s'éloigne des bords.

D = -D;

% Suppression des minima non significatifs
D=imhmin(D,2);

L = watershed(D);

% Les points à 0 sont la ligne de partage des eaux, les labels représentent
% les bassins versants.
rgb = label2rgb(L);
figure(3), imshow(rgb), title('Watershed avec les bassins versants')

% Mise à 0 des points de la LPE dans l'image initiale
IBWW=IBW;
IBWW(L==0)=0;
figure(4),imshow(IBWW);
title('Watershed transform with euclidean distance');
```

fonction distance



Watershed avec les bassins versants





1.3. In this approach, what is the main parameter ?

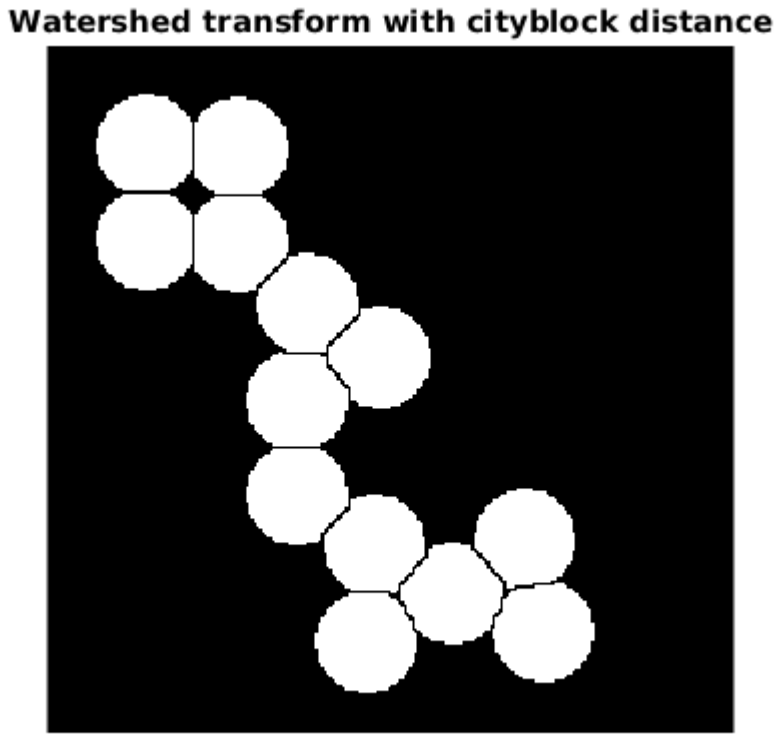
Modify it to observe its influence.

Answer : the distance function

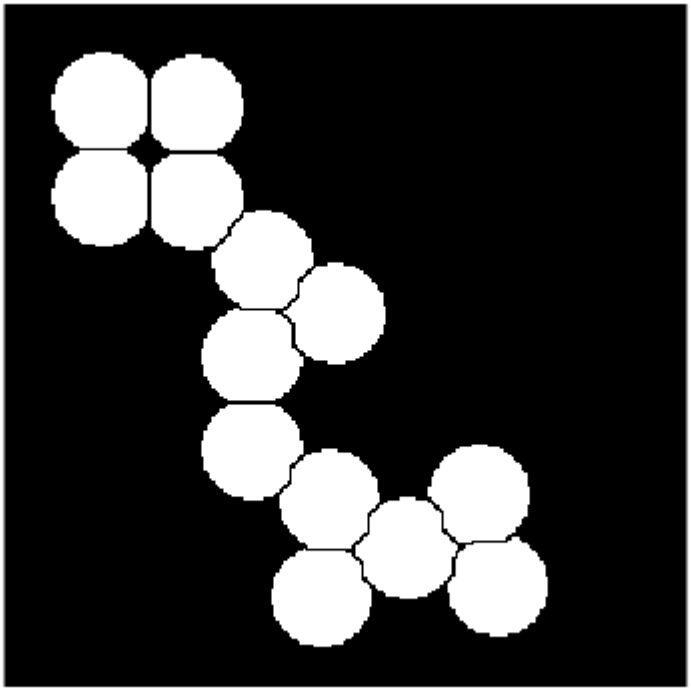
```
D = bwdist(~IBW,'cityblock');
D = -D;
D=imhmin(D,2);
L = watershed(D);
IBWW=IBW;
IBWW(L==0)=0;
figure(5), imshow(IBWW)
title('Watershed transform with cityblock distance')

D = bwdist(~IBW,'chessboard');
D = -D;
D=imhmin(D,2);
L = watershed(D);
IBWW=IBW;
IBWW(L==0)=0;
figure(6), imshow(IBWW)
title('Watershed transform with chessboard distance')

D = bwdist(~IBW,'quasi-euclidean');
```

```
D = -D;
D=imhmin(D,2);
L = watershed(D);
IBWW=IBW;
IBWW(L==0)=0;
figure(7), imshow(IBWW)
title('Watershed transform with quasi-euclidean distance')
```

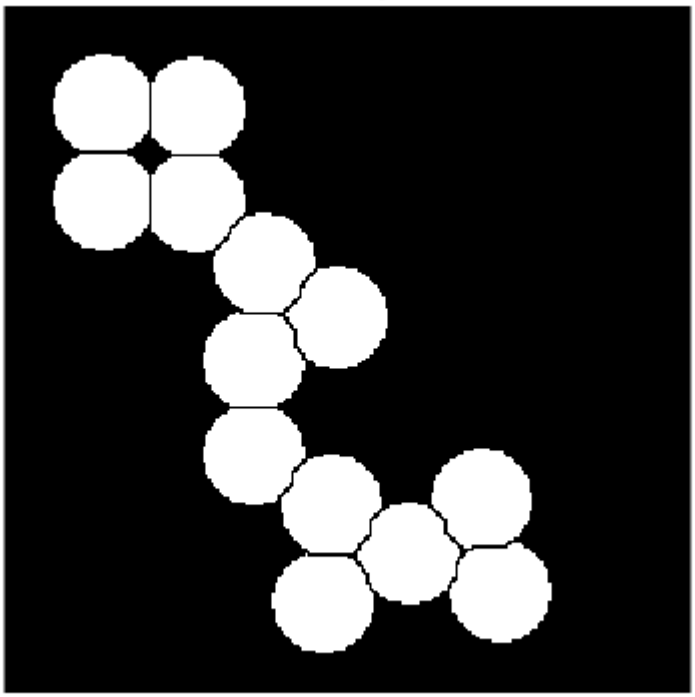
Observations : On note qui'il y a des petites différences dans les résultats en fonction de la distance utilisée



Watershed transform with chessboard distance



Watershed transform with quasi-euclidean distance



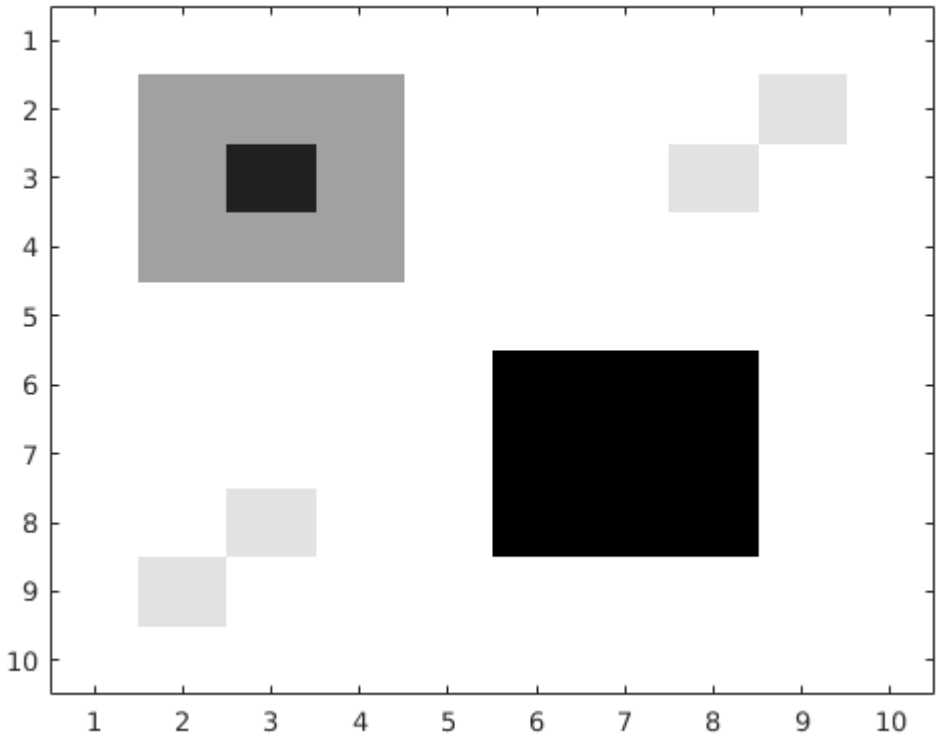
PART 2 - Minima imposition

You can emphasize specific minima (dark objects) in an image using the imimposemin function. The imimposemin function uses morphological reconstruction to eliminate all minima from the image except the minima you specify.

To illustrate the process of imposing a minimum, this code creates a simple image containing two primary regional minima and several other regional minima.

```
clear all;
close all;

mask = uint8(10*ones(10,10));
mask(6:8,6:8) = 2;
mask(2:4,2:4) = 7;
mask(3,3) = 3;
mask(2,9) = 9;
mask(3,8) = 9;
mask(9,2) = 9;
mask(8,3) = 9

figure(1),imagesc(mask),colormap('gray');
```

```
mask =

   10   10   10   10   10   10   10   10   10   10
   10    7    7    7   10   10   10   10    9   10
   10    7    3    7   10   10   10    9   10   10
   10    7    7    7   10   10   10   10   10   10
   10   10   10   10   10   10   10   10   10   10
   10   10   10   10   10    2    2    2   10   10
   10   10   10   10   10    2    2    2   10   10
   10   10    9   10   10    2    2    2   10   10
   10    9   10   10   10   10   10   10   10   10
   10   10   10   10   10   10   10   10   10   10
```



2.1. Creating a Marker Image

To obtain an image that emphasizes the two deepest minima and removes all others, create a marker image that pinpoints the two minima of interest. You can create the marker image by explicitly setting certain pixels to specific values or by using other morphological functions to extract the features you want to emphasize in the mask image (ultimate erosion, minr, maxr, ..).

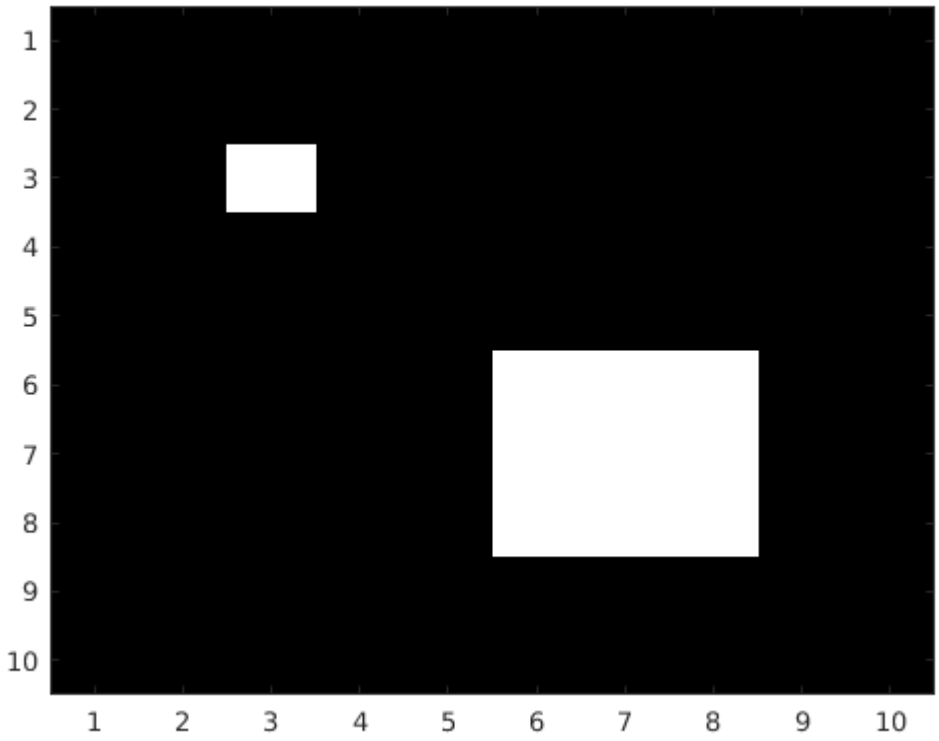
-> Propose a method to extract the two deepest minima

This example uses imextendedmin to get a binary image that shows the locations of the two deepest minima.

```
marker = imextendedmin(mask,2)
figure(2),imagesc(marker),colormap('gray');
```

```
marker =

     0     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     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     1     1     1     0     0
     0     0     0     0     0     1     1     1     0     0
     0     0     0     0     0     1     1     1     0     0
```

```
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
```



2.2. Applying the Marker Image to the Mask

a) Now use imimposemin to create new minima in the mask image at the points specified by the marker image.

Note how imimposemin sets the values of pixels specified by the marker image to the lowest value supported by the datatype (0 for uint8 values).

```
I = imimposemin(mask,marker)
figure(3),imagesc(I),colormap('gray');
```

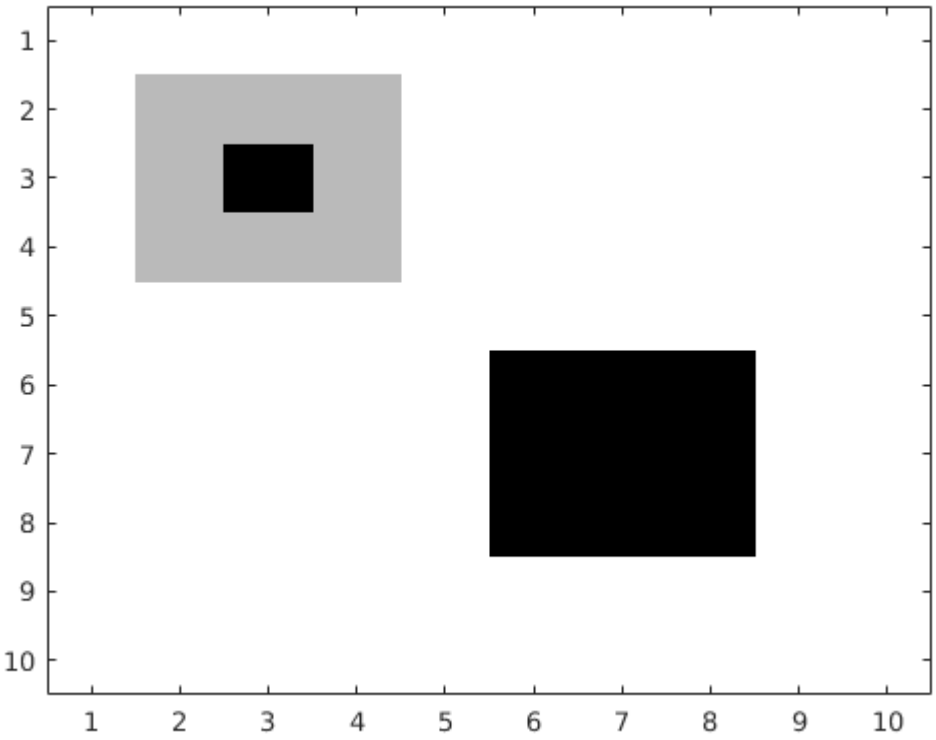
b) Comment the result

imimposemin also changes the values of all the other pixels in the image to eliminate the other minima.

Remark : you can modify the test image values to understand the method.

```
I =

   11   11   11   11   11   11   11   11   11   11
   11    8    8    8   11   11   11   11   11   11
   11    8    0    8   11   11   11   11   11   11
   11    8    8    8   11   11   11   11   11   11
   11   11   11   11   11   11   11   11   11   11
```

```
11   11   11   11   11    0    0    0   11   11
11   11   11   11   11    0    0    0   11   11
11   11   11   11   11    0    0    0   11   11
11   11   11   11   11   11   11   11   11   11
11   11   11   11   11   11   11   11   11   11
```



PART 3 - Marker-controlled Watershed

If the imaging conditions are not optimal or the image is cluttered, thresholding does not produce acceptable segmentation.

3.1.Segmentation of vertical stripes

3.1.1. Open the image 'fabricTexture.jpg' and threshold it.

```
clear all;
close all;

I=imread('fabricTexture.jpg');
I=rgb2gray(I);
figure(1), imshow(I), title('Image originale')

I2=I;
level = graythresh(I);
```

```
IBW = im2bw(I,level);
figure(2), imshow(IBW), title('Image seuillée')
```

Comment the result : les stripes ne sont pas séparées correctement par seuillage.

3.1.2. Apply a marker-controlled Watershed segmentation to detect white vertical stripes.

The goal is to obtain a result image similar to the given fabricTextureSeg.jpg

```
Ires=imread('fabricTextureSeg.jpg');
figure(3),imshow(Ires), title('Segmentation à obtenir')
```

Hint : read the Matlab demo : Image Processing-> Image Segmentation -> Marker-controlled watershed segmentation.

3.1.2.1. Definition of internal markers

Objectif : on veut imposer des minima entre les bandes pour que les eaux se séparent au niveau des centres des bandes blanches.

```
Idil=imdilate(I2, strel('rectangle',[40 10]));
figure(4),imshow(Idil),title('Dilatation par un rectangle vertical')
```

On garde comme marqueurs les minimas qui ont une amplitude supérieure à 100.

```
Idilemin = imextendedmin(Idil,100);
figure(5),imshow(Idilemin), title('Marqueurs entre les stripes')
```

Ces minima régionaux nous servent de marqueurs pour la ligne de partage des eaux. Remarque si on prend un seuil trop bas –> on va récupérer des minima régionaux, qui correspondent à des petites zones du fond des vallées.

3.1.2.2. Definition of external markers

External markers are obtained through a watershed on the original image after minima imposition with internal markers.

```
L=watershed(imimposemin(I2,Idilemin));
figure(6), imshow(L==0), title('Marqueurs sur les stripes')
```

Fusion des 2 ensembles de marqueurs

```
marker=(L==0) | Idilemin;
figure(7), imshow(marker), title('Marqueurs fusionnés');
```

3.1.2.3. Watersehd segmentation

Calcul du gradient morphologique

```
el=strel('disk',1);
grad=imdilate(I2,el)-imerode(I2,el);
figure(8), imshow(grad), title('Gradient morphologique');
```

Watershed final sur le gradient en imposant les min à partir des marqueurs internes et externes fusionnés

```
L2=watershed(imimposemin(grad,marker));
figure(9), imshow(L2==0), title('LPE sur le gradient avec marqueurs')
```

```
% Mise à 0 des points détectés
I(L2==0)=0;

% modification de la lut
map=gray(256);
map(1,:)=[1 0 0];

figure(10)
imshow(I), title('Segmentation finale')
colormap(map)
```

Image originale

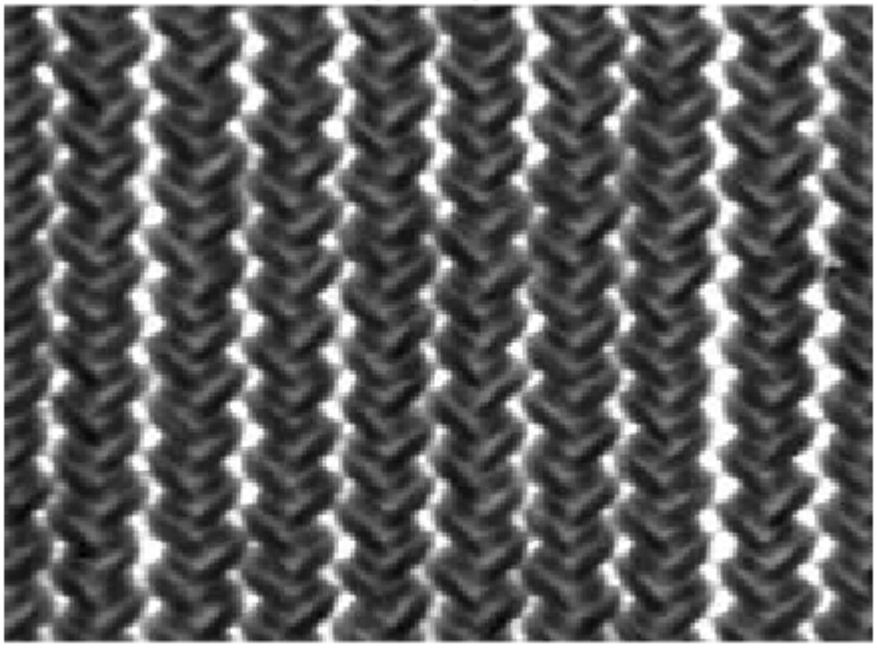
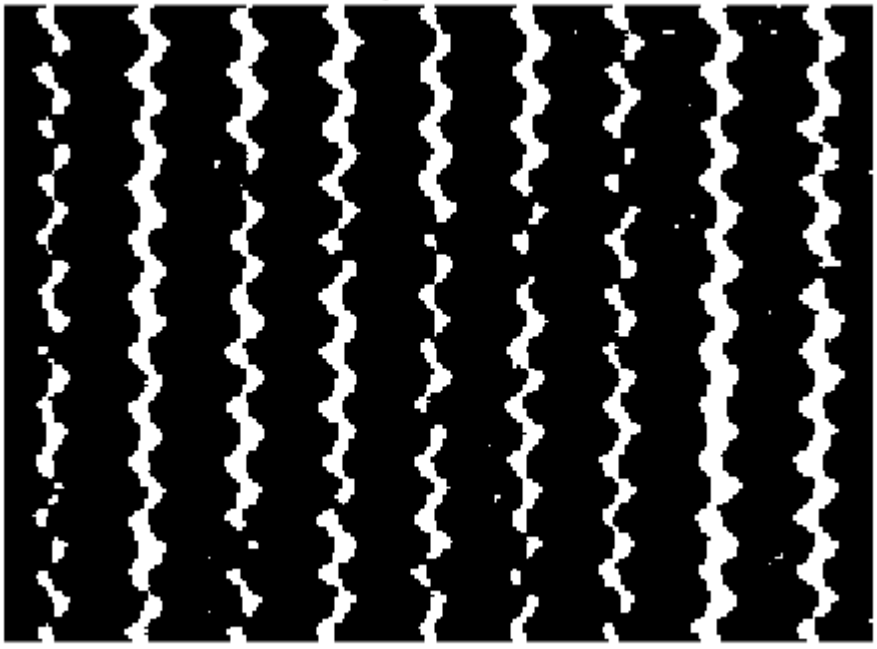
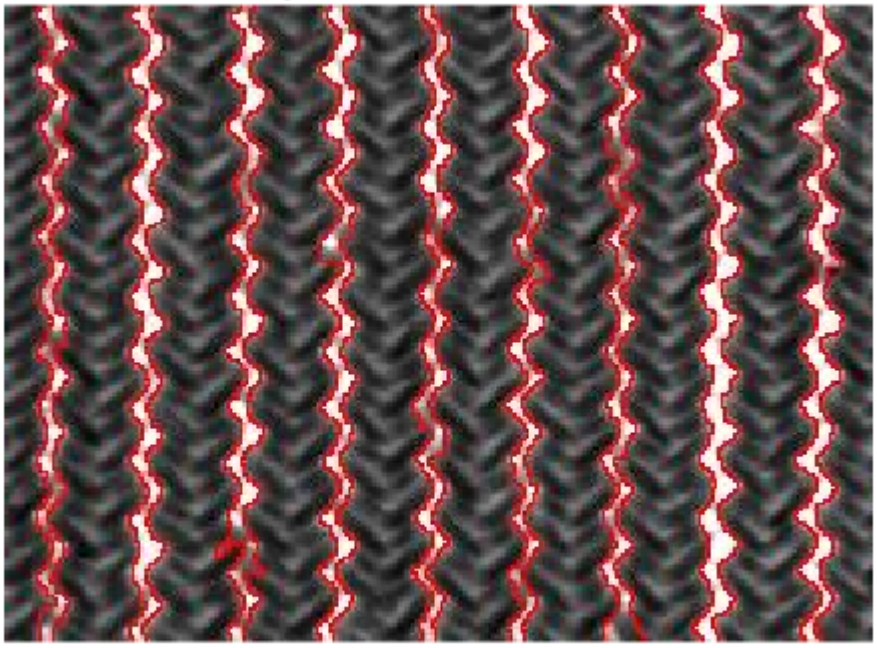


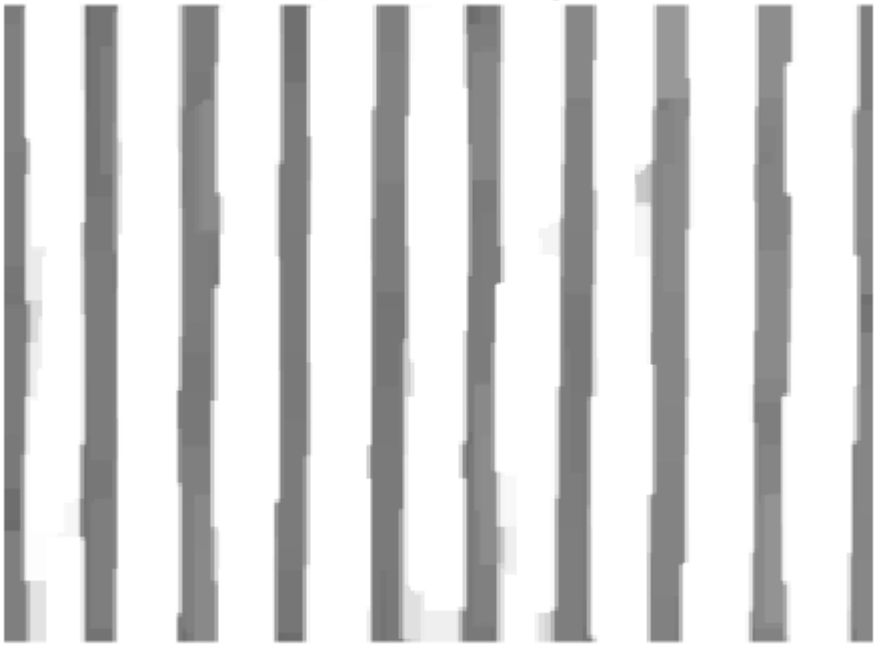
Image seuillée



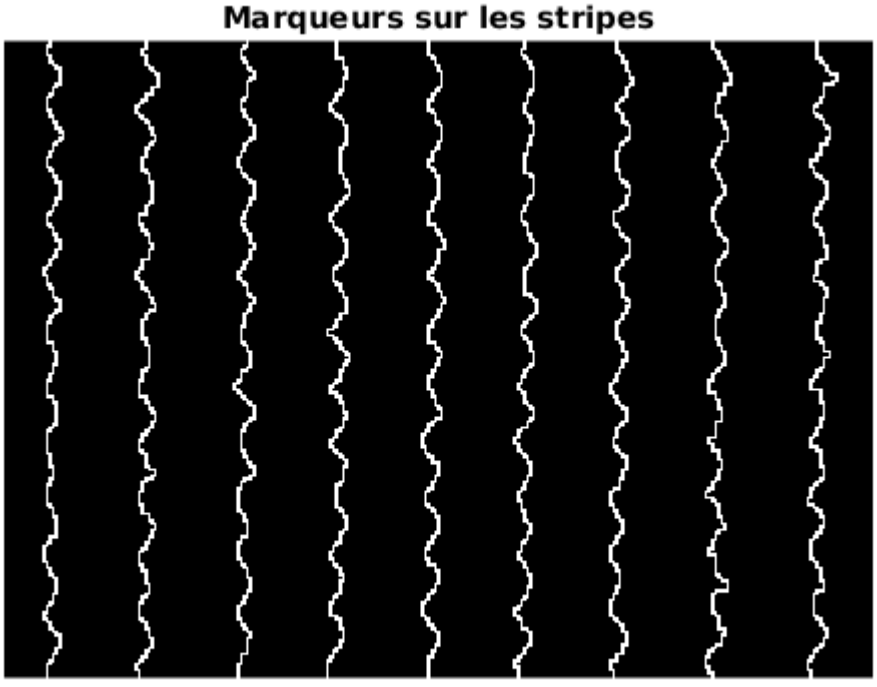
Segmentation à obtenir



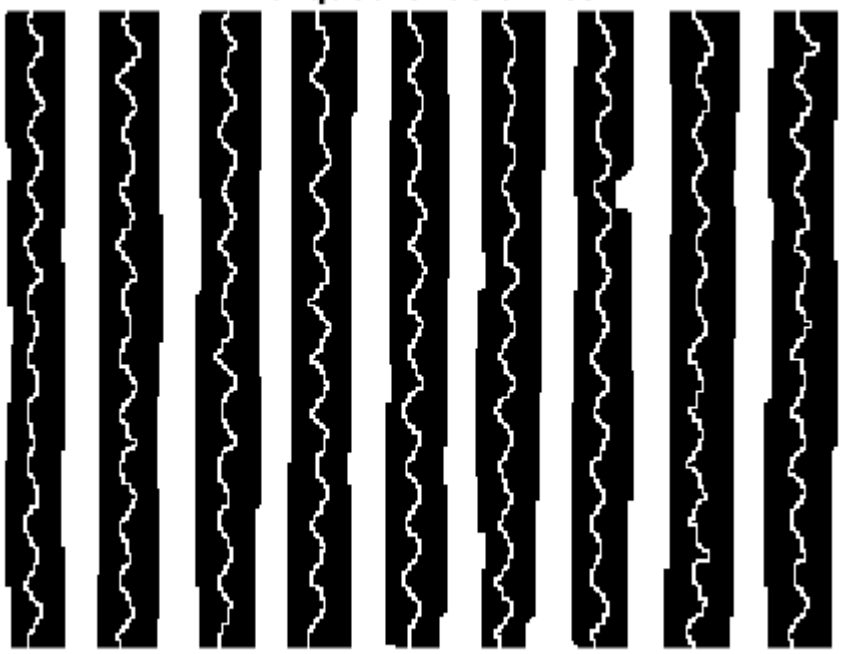
Dilatation par un rectangle vertical




Marqueurs entre les stripes

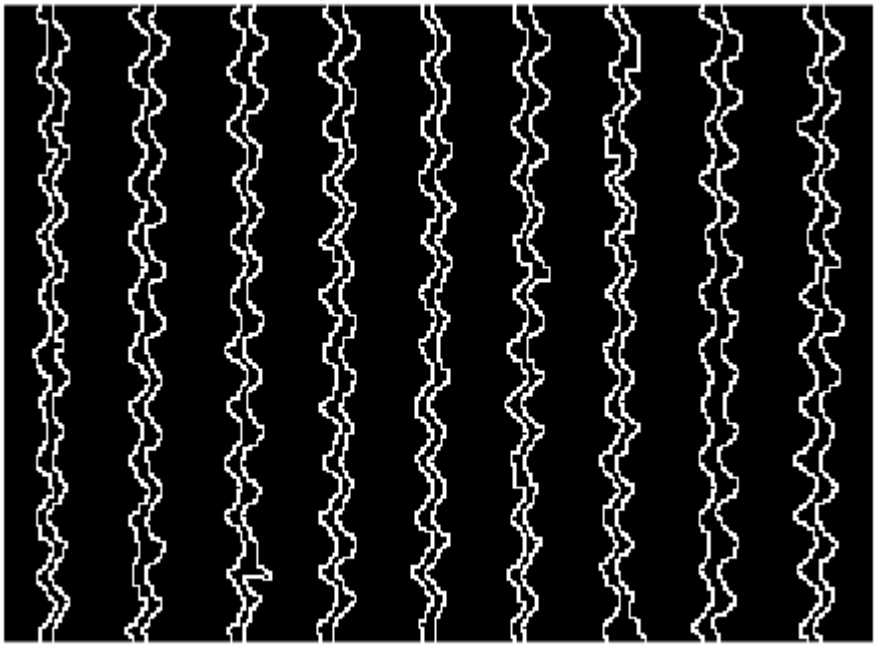

Marqueurs sur les stripes

Marqueurs fusionnés

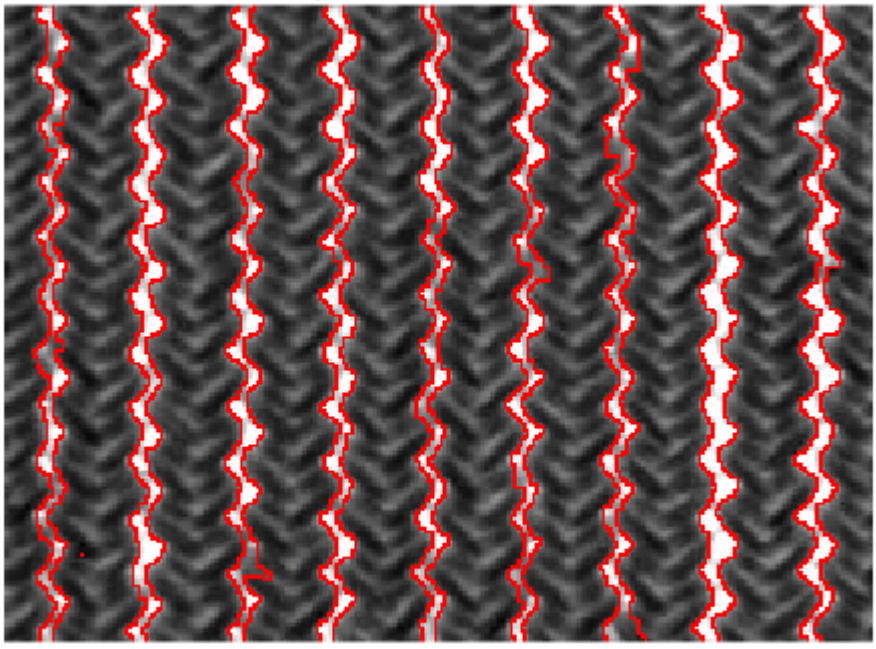


Gradient morphologique

LPE sur le gradient avec marqueurs



Segmentation finale



3.2. Segmentation of the beef image

3.2.1. Open the image 'boeuf.jpg'

```
clear all
close all
I=rgb2gray(imread('boeuf.jpg'));
figure(1), imshow(I), title('Image initiale')

level = graythresh(I);
IBW1 = im2bw(I,level);
figure(2), imshow(IBW1), title('Image seuillée')
```

Comment the result : impossible de détecter la zone centrale.

3.2.2. Apply a marker-controlled Watershed segmentation to detect the steak.

The goal is to obtain a result image similar to the given boeufSeg.jpg

```
Ires=imread('boeufSeg.jpg');
figure(3),imshow(Ires), title('Segmentation à obtenir')
```

3.2.2.1. Definition of internal markers

Le principe est de segmenter grossièrement l'image et de garder une zone centrale servant de marqueur.

```
% Segmentation de l'image
IBW = im2bw(I,0.1); % avec 0.01, on voit du bruit autour du contour, 0.1 OK
figure(4);imshow(IBW), title('Pré-segmentation')

IBWcl = imclose(IBW,strel('disk',3));
figure(5);imshow(IBWcl), title('Fermeture')

IBWint = imerode(IBWcl,strel('disk',80));
figure(6);imshow(IBWint), title('Erosion pour garder le centre');
```

3.2.2.2. Definition of external markers

On commence par détecter un marqueur des bords externes qui est assez éloigné de la zone à segmenter. Pour mieux positionner le contours lors de la segmentation finale, on réalise une segmentation fine des zones externes par une première LPE sur l'image originale.

```
IBWbord = IBWcl-imerode(IBWcl,strel('disk',13));
figure(7);imshow(IBWbord), title('Marqueurs bord externe')

% Marqueurs plus précis obtenus par une LPE sur l'image originale
IBWext = watershed(imimposemin(I,IBWint+IBWbord))==0;
figure(8);imshow(IBWext), title('Marqueurs externes précis')

% Marqueurs fusionnés
Imarker=IBWint | IBWext;
figure(9);imshow(Imarker), title('Marqueurs fusionnés')
```

3.2.2.3. Watersehd segmentation

Calcul de l'image de gradient morphologique

Observe the gradient image. The contour is not closed !! It's possible to filter the image to improve its content.

```
% Pre-processing : on enlève les filaments gris trop petits
I2=imclose(I,strel('disk',4));

% Morphological gradient (on peut utiliser aussi les autres gradients
% morphologiques)
Gmag1=imdilate(I2,strel('disk',2))-I2;
figure(10);imshow(Gmag1,[]), title('Morphological gradient');

% Linear gradient par filtre de sobel
h=fspecial('sobel');
gx=imfilter( double(I2), h , 'replicate' );
gy=imfilter( double(I2), h' , 'replicate' );
Gmag2=sqrt(gx.^2 + gy.^2);
figure(11);imshow(Gmag2,[]), title('Linear gradient');

% Watershed segmentation for the two gradient with imposed marker image
L1=watershed(imimposemin(Gmag1,Imarker));
L2=watershed(imimposemin(Gmag2,Imarker));

% Watershed plotted on the original image
IF1=I-1;
IF1(L1==0)=255;
IF2=I-1;
IF2(L2==0)=255;

% Modification de la lut
map=gray(256);
map(256,:)=[1 0 0];

figure(12)
imshow(IF1), title('Watershed with morphological gradient');
colormap(map)

figure(13)
imshow(IF2), title('Watershed with linear gradient');
colormap(map)
```

Remarques : le gradient linéaire donne un résultat plus précis. Le prétraitement par fermeture permet de supprimer la zone protubérante en bas à droite. La taille optimale pour la fermeture est de 4. Une valeur plus élevée supprime des parties à segmenter.

Image initiale

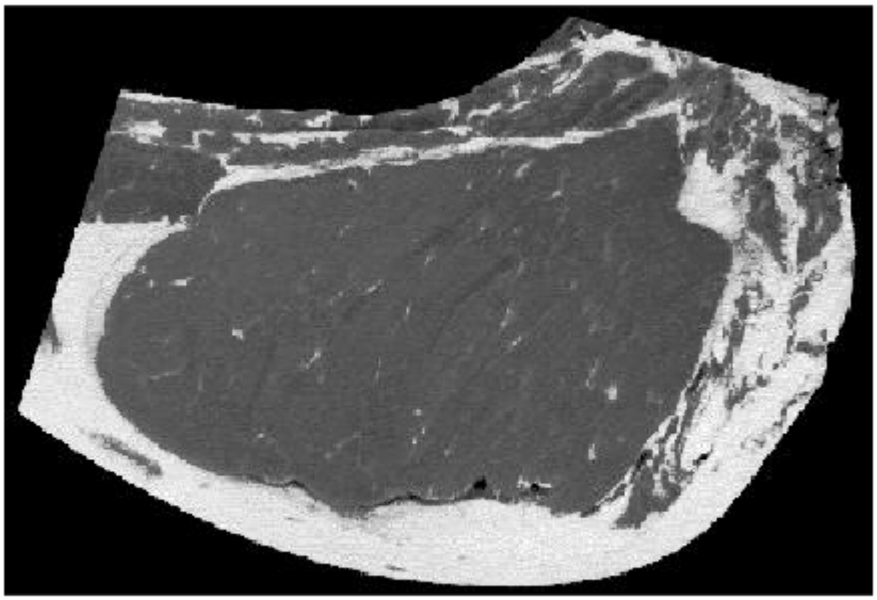
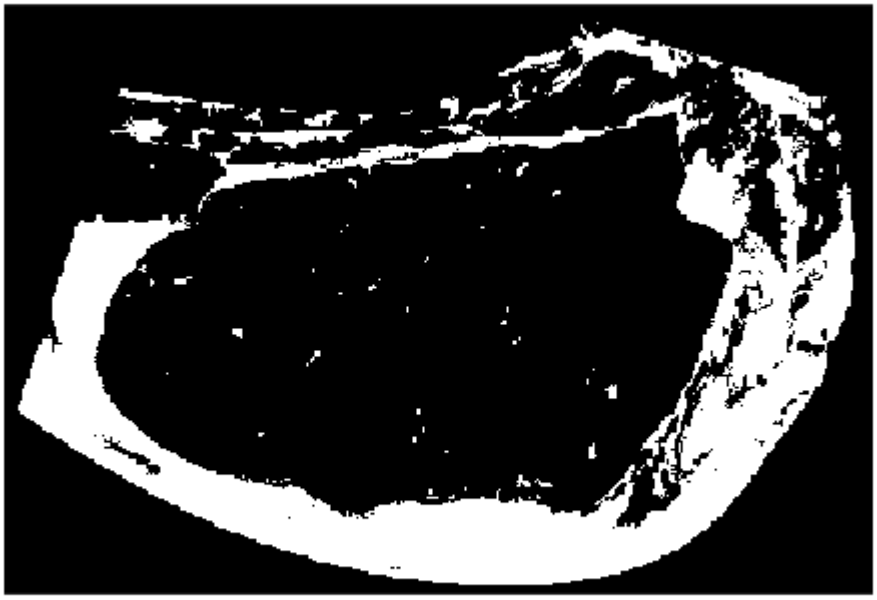
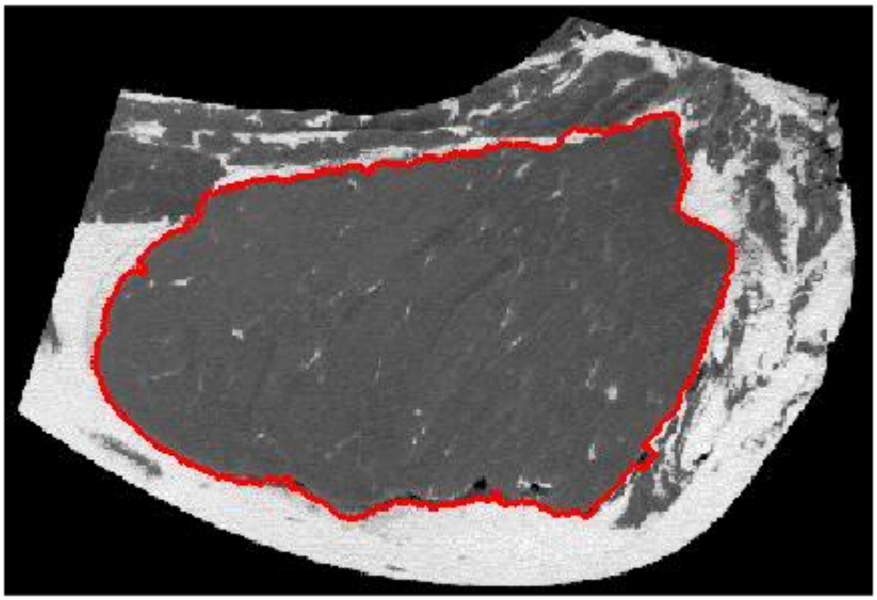


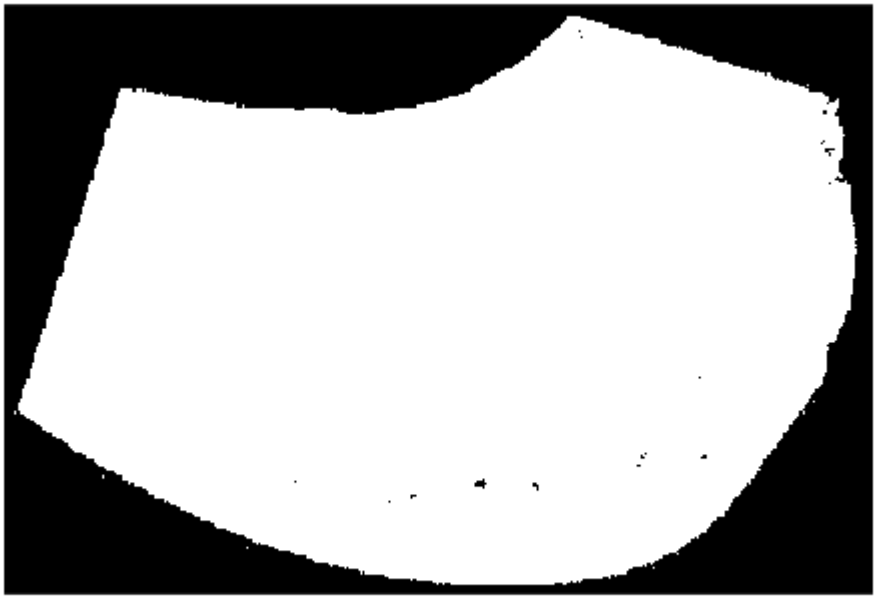
Image seuillée



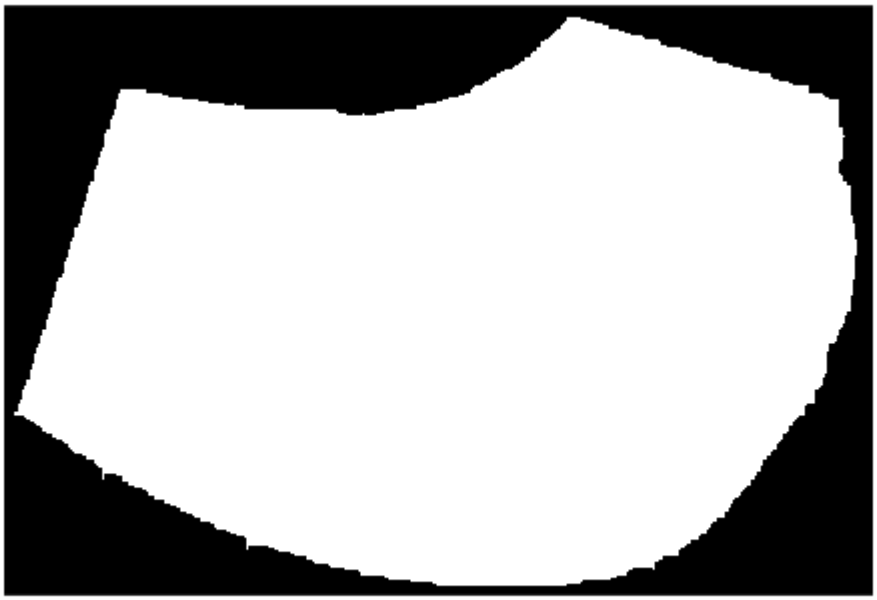
Segmentation à obtenir



Pré-segmentation

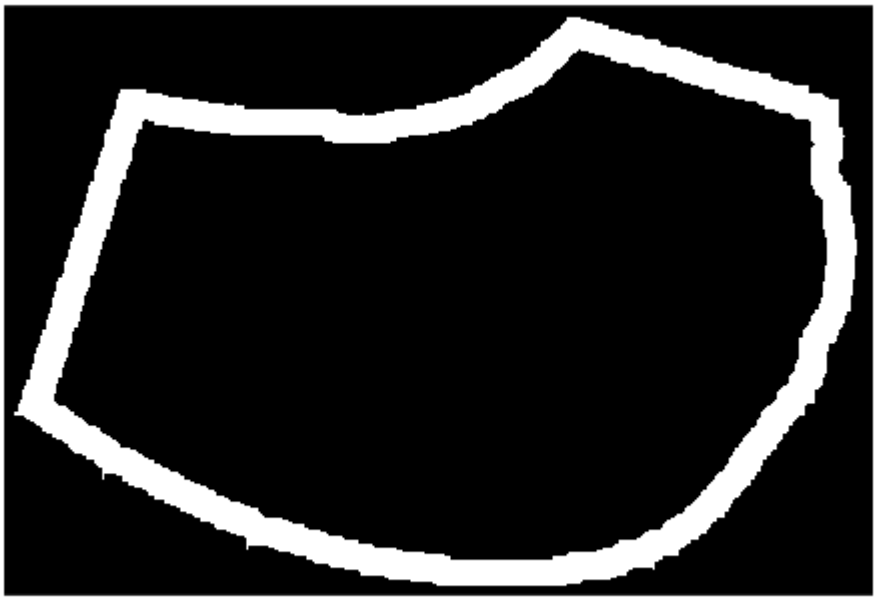


Fermeture



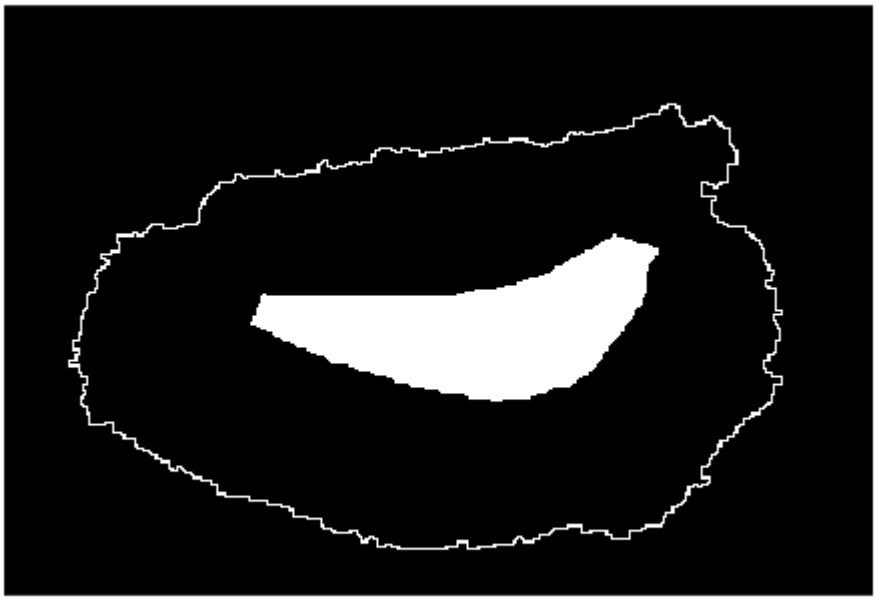
Erosion pour garder le centre

Marqueurs bord externe

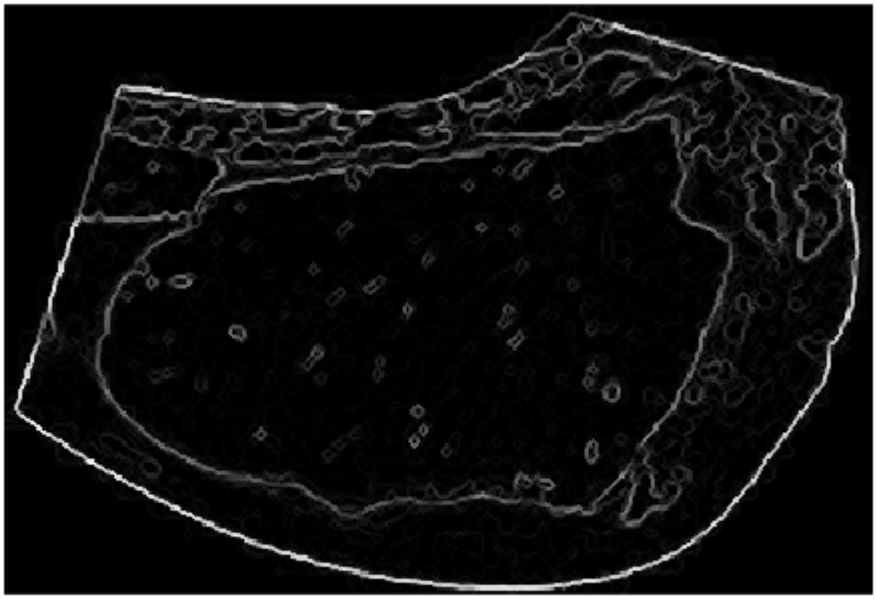


Marqueurs externes précis

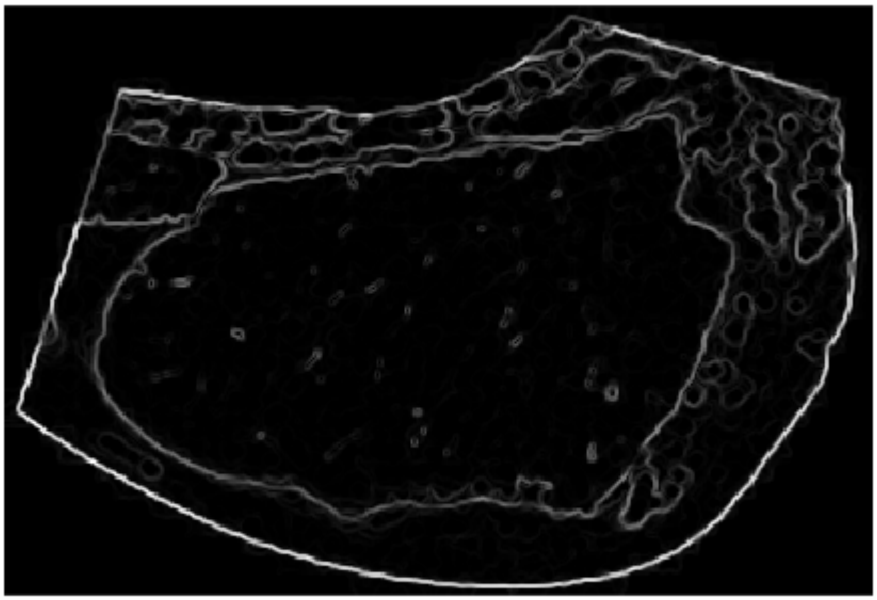
Marqueurs fusionnés



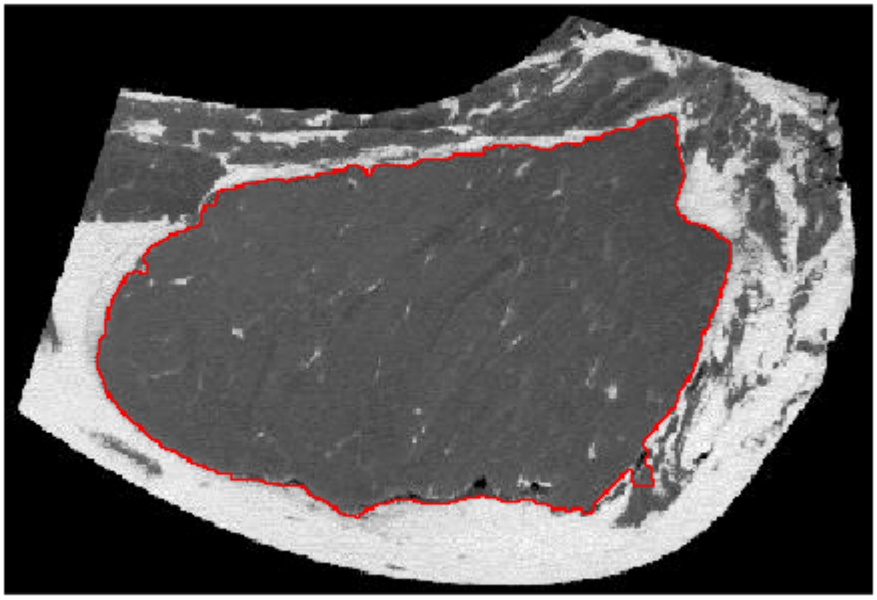
Morphological gradient

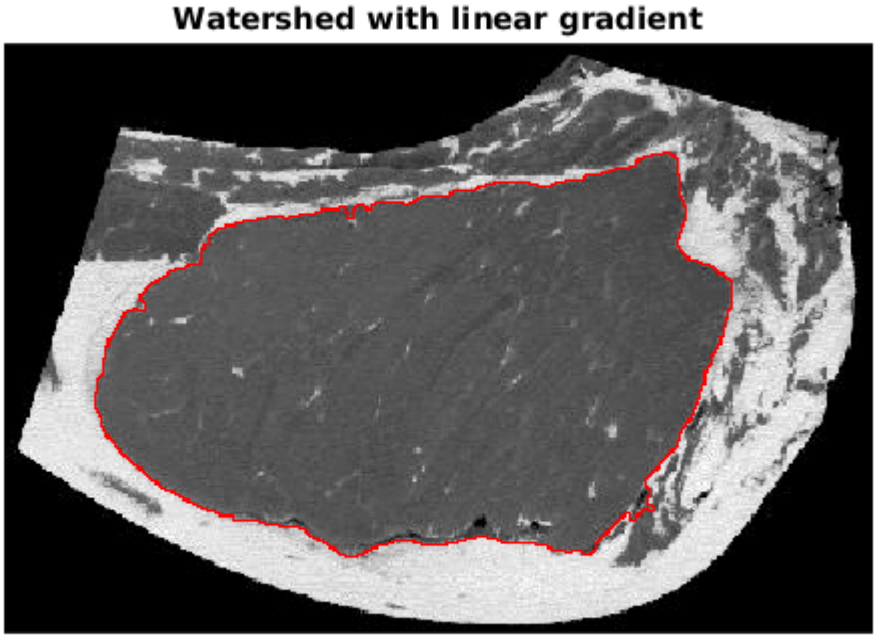


Linear gradient



Watershed with morphological gradient





Published with MATLAB® R2015b