Traitement d'images

TP noté

Matthéo FERTE CIPA5
27 Novembre 2024

Pour faire cela, plusieurs étapes peuvent être réalisées :

1. Lire et afficher l'image "image.png". Quel est le type de cette image ? Quelle est la taille de cette image ?
2. Afficher l'histogramme de cette image. Que peut-on déduire ?
3. Binariser cette image de façon à extraire les différents objets de l'image (séparer les objets du fond). Afficher l'image obtenue.
4. Raffiner et améliorer l'image obtenue en appliquant les outils de la morphologie mathématique.
    a. Pour chaque opération morphologique réalisée, préciser comment et pourquoi vous l'avez utilisée ? Quelle est la taille de l'image obtenue ? C'est quoi son type ?
5. Faire une analyse des composants connectés. Indiquer le nombre total de tous les objets détectés dans l'image.
    a. Pour chacun des objets étiquetés, calculer son aire "A" et son périmètre "P". En effet, un objet est considéré comme un cercle ou un disque si :
    $$0.8 \leq \frac{4\pi \times A}{P^2} \leq 1.2$$
6. Ainsi les objets qui valident cette condition sont à conserver dans l'image binaire. Afficher la nouvelle image binaire qui contient uniquement les objets conservés.
7. Une dernière étape consiste à afficher en vert dans l'image originale, les rectangles qui englobent les cercles et les disques détectés.

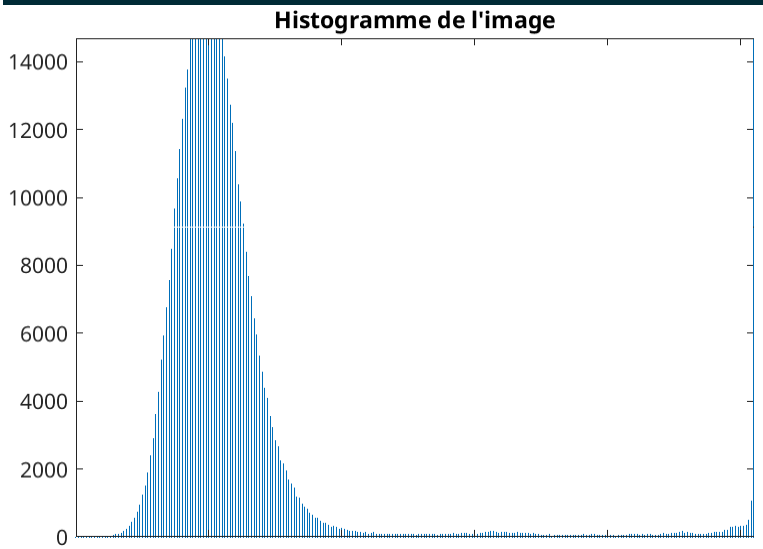
Question 1 :

```
% Charger l'image "image.png"
img = imread("image.png","png");
% Afficher l'image originale
figure;
imshow(img);
title('Image originale');
% Déterminer le type de l'image
disp('Type de l''image :');
disp(class(img)); % Affiche le type (uint8, double, etc.)
```

Le type de l'image de l'image est ***UINT8*** et la taille est 512x384.

Question 2 :

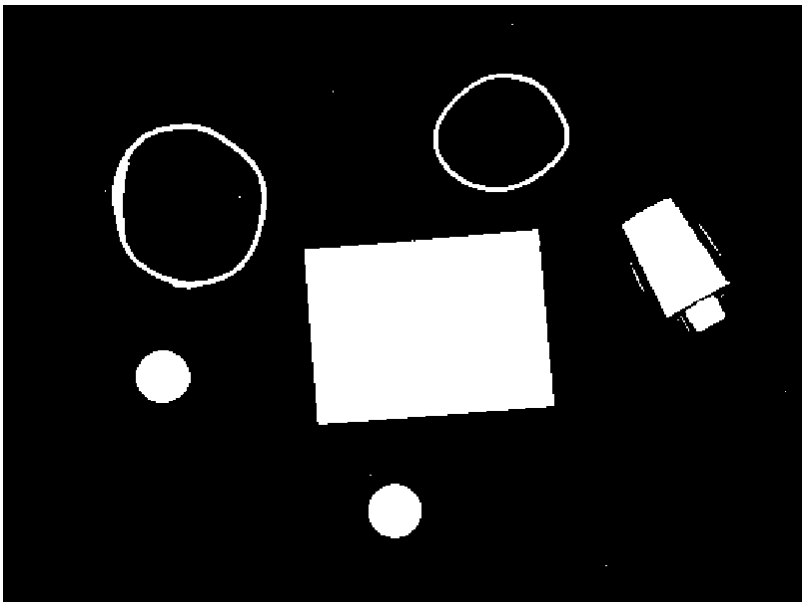
```
% Afficher l'histogramme de l'image
figure;
imhist(img);
title('Histogramme de l''image');
```



On peut en déduire une forte présence de pixels sombres et un pic de pixels très clairs. Il doit donc y avoir un fort contraste entre les objets de l'image.

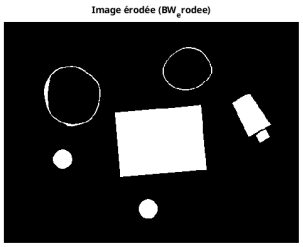
Question 3 :

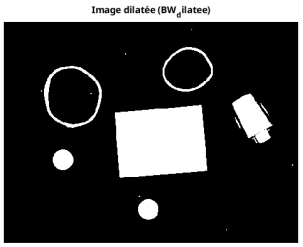
```
% Convertir en niveaux de gris si l'image est en couleur
if size(img, 3) == 3
  image_gris = rgb2gray(img);
else
  image_gris = img;
end
% Binariser l'image avec un seuil automatique (méthode d'Otsu)
seuil = graythresh(image_gris); % Calcule le seuil optimal
image_binaire = imbinarize(image_gris, seuil);
% Afficher l'image binaire
figure;
imshow(image_binaire);
title('Image binaire');
```

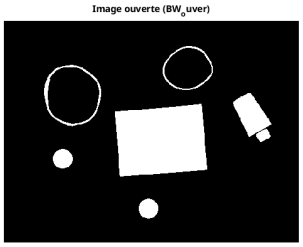


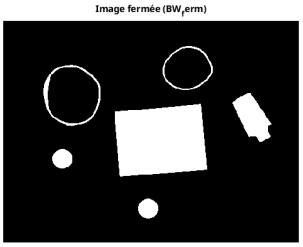
Question 4 :

```
% Appliquer les opérateurs de morphologie mathématique
% 1. Érosion
se1 = strel('square', 2); % Élément structurant carré de taille 2
BW_erodee = imerode(image_binaire, se1);
% 2. Dilatation
BW_dilatee = imdilate(image_binaire, se1);
% 3. Ouverture
BW_ouver = imopen(image_binaire, se1);
% 4. Fermeture sur l'image ouverte avec un élément structurant de taille 10
se2 = strel('square', 10); % Élément structurant carré de taille 10
BW_ferm = imclose(BW_ouver, se2);
% Afficher les résultats intermédiaires
figure;
subplot(2, 2, 1); imshow(BW_erodee); title('Image érodée (BW_erodee)');
subplot(2, 2, 2); imshow(BW_dilatee); title('Image dilatée (BW_dilatee)');
subplot(2, 2, 3); imshow(BW_ouver); title('Image ouverte (BW_ouver)');
subplot(2, 2, 4); imshow(BW_ferm); title('Image fermée (BW_ferm)');
```









	Érodée	Dilatée	Ouverte	Fermée
Pourquoi ?	Éroder permet de retirer le bruit en retirant la couche extérieure de pixels des objets de l'image.	La dilatation permet, en agrandissant d'un pixel tous les objets, de limiter la discontinuité, les trous et les artefacts.	L'ouverture est utilisée pour supprimer les petits objets ou le bruit isolé qui pourraient être présents dans l'image.	La fermeture comble les petits trous et les espaces à l'intérieur des objets, ce qui est important pour regrouper des parties d'objets disjointes ou éliminer des discontinuités.
Taille	384x512	384x512	384x512	384x512
Type	logical	logical	logical	logical

Question 5 :

```
% Étiquetage des objets
[img_label, num_objects] = bwlabel(BW_ferm);
figure;
imshow(img_label);
% Afficher le nombre d'objets détectés
fprintf('Nombre d''objets détectés : %d\n', num_objects);
```

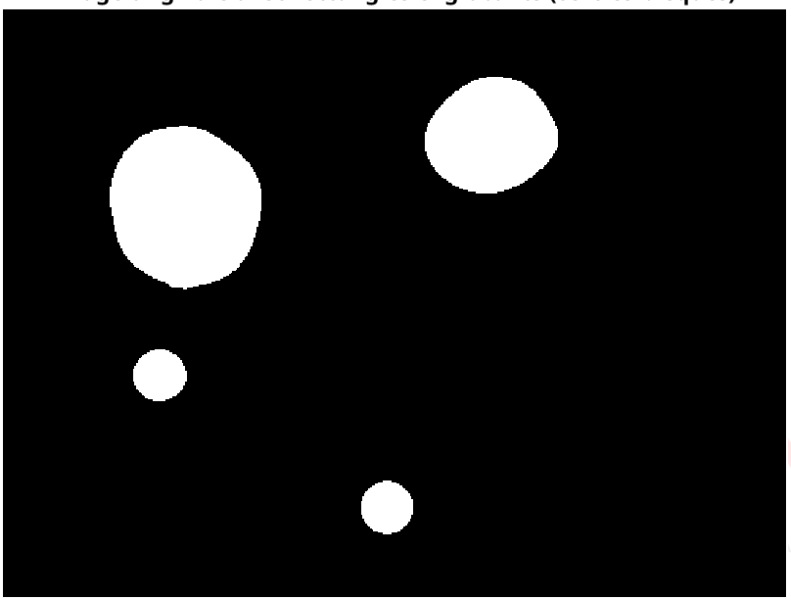
Console : "*Nombre d'objets détectés : 6*"

Question 6 :

```
img_fill = imfill(img_label,"holes");
% Calculer les propriétés géométriques
props = regionprops(img_fill, 'Area', 'Perimeter');
% Créer une nouvelle image binaire pour conserver uniquement les
cercles/disques
nouvelle_image = false(size(image_binaire));
for k = 1:num_objects
  % Récupérer l'aire et le périmètre de l'objet
  A = props(k).Area;
  P = props(k).Perimeter;

  % Vérifier la condition : 0.8 <= (7.4 * pi * A) / (P^2) <= 1.2
  if P > 0 % Éviter les divisions par zéro
    critere = (4 * pi * A) / (P^2);
    if critere >= 0.8 && critere <= 1.2
      % Conserver cet objet
      nouvelle_image = nouvelle_image | (img_fill == k);
    end
  end
end
% Afficher l'image binaire
figure;
imshow(nouvelle_image);
title('Image contenant uniquement les cercles/disques');
```

Image originale avec rectangles englobants (cercles/disques)



Question 7 :

```
% Créer une nouvelle image binaire pour conserver uniquement les cercles/disques
nouvelle_image = false(size(image_binaire));
bounding_boxes = []; % Liste des bounding boxes à afficher

for k = 1:num_objects
    % Récupérer l'aire et le périmètre de l'objet
    A = props(k).Area;
    P = props(k).Perimeter;

    % Vérifier la condition : 0.8 <= (7.4 * pi * A) / (P^2) <= 1.2
    if P > 0 % Éviter les divisions par zéro
        critere = (4 * pi * A) / (P^2);
        if critere >= 0.8 && critere <= 1.2
            % Conserver cet objet
            nouvelle_image = nouvelle_image | (img_fill == k);
            bounding_boxes = [bounding_boxes; props(k).BoundingBox];
        end
    end
end
```

```
% Convertir l'image originale en RGB pour superposition
if size(img, 3) == 1
  image_originale_rgb = cat(3, img, img, img);
else
  image_originale_rgb = img;
end
% Dessiner les rectangles englobants en vert sur l'image originale
for i = 1:size(bounding_boxes, 1)
  rectangle_position = bounding_boxes(i, :); % Position [x, y, largeur, hauteur]
  image_originale_rgb = insertShape(image_originale_rgb, 'Rectangle',
rectangle_position, ...
                                    'Color', 'green', 'LineWidth', 3);
end
% Afficher l'image avec les rectangles englobants
figure;
imshow(image_originale_rgb);
title('Image originale avec rectangles englobants (cercles/disques)');
```