DURET Guillaume GR IMA
SCHIEB Antoine

Transphormée de Hough :

2018-2019

INTRODUCTION

Nous voulons dans ce tp utilisé la méthode de Hough qui permet de détecter sur une image les lignes droites. Dans une seconde partie nous allons détecter des cercles.

I) Detection de droites

1) Obtention du gradient de l'image

Avant d'utiliser la méthode de Hough nous devons obtenir les contours de l'image afin de bien appliquer la méthode sur ceux-ci.

```
% tp HOUGH
close all;
clear;
clear variables;
%%lecture de l'image
I=rgb2gray(im2double(imread('hello.png')));
figure(1);
imshow(I);
title('image d''origine');


[Gmag, ~] = imgradient(I,'prewitt');
figure(2);
imshow(Gmag);
% seuillage de Gmax pour Ã©limiiner le bruit
I_grad_seuil = im2bw(Gmag,0.76);
figure(3);
imshow(I_grad_seuil);
title('image du gradient');

%fermeture du seuil
figure(4);
%%SE=strel('square',4); %antoine
SE=strel('square',3); %guillaume

% pour hello.png
I_seuil_propre=I_grad_seuil;

% pour circuit.tif
%I_seuil_propre = imerode(I_grad_seuil,SE);
%imshow(I_seuil_propre,[]);
```

Pour l’image de « HELLO » nous obtenons le résultat suivant :

image d'origine

image du gradient

Et pour l’image d’un circuit, nous avons dû calculer le gradient, seuillé celui-ci et enfin réaliser des érosions et ouverture.

image d'origine

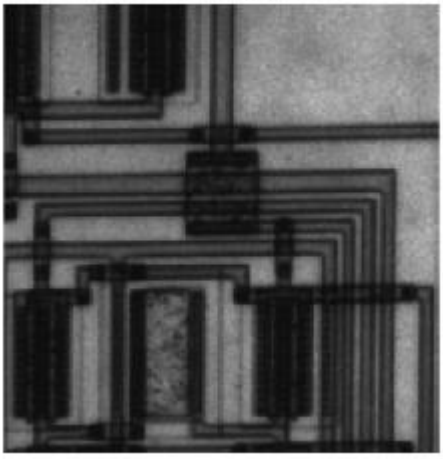


image du gradient

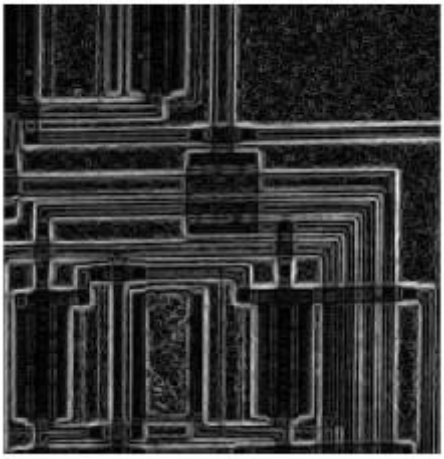
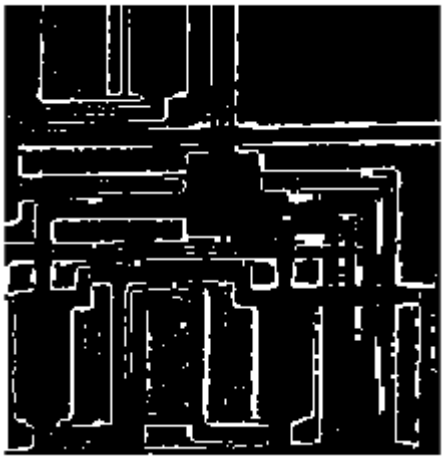


image dues contours finaux



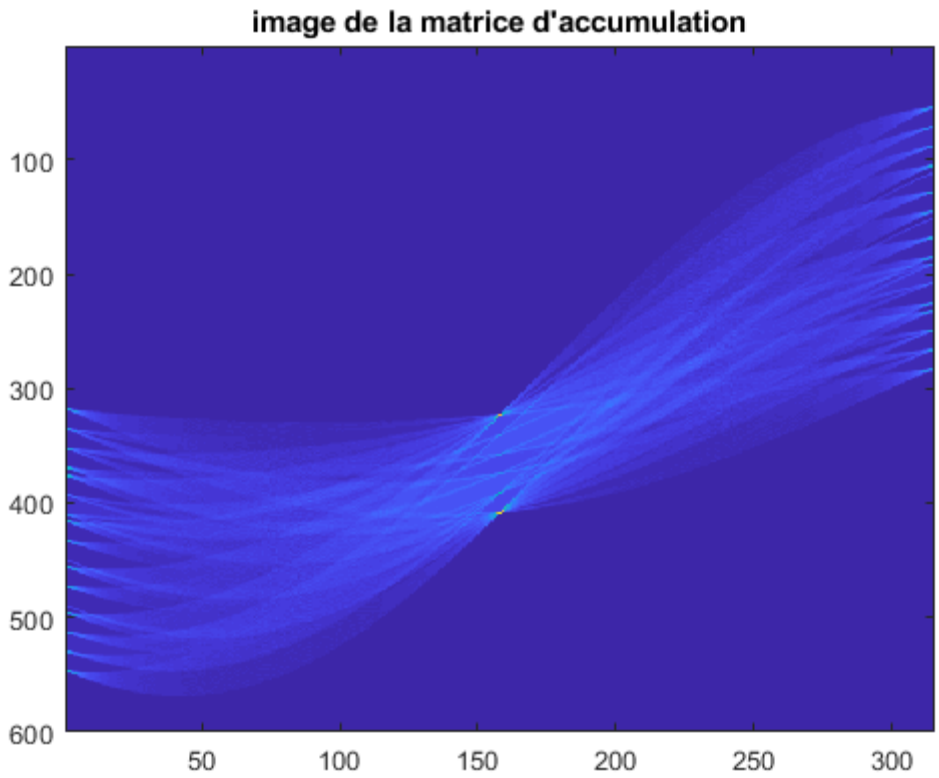
2) Méthode de Hough

Nous appliquons donc sur les contours de l'image d'origine les différentes étapes de la méthode de Hough

Tout d'abord nous déterminons la matrice d'accumulation qui permet de détecter dans le repère (ro, thêta) les droites de l'image.

```
Ro_max=ceil(sqrt(size(I,1)^2+size(I,2)^2));

dtheta=0.01;

theta=0:dtheta:pi-dtheta;

H = zeros(ceil(2*Ro_max),ceil(pi/dtheta));

%boucle sur les points du contour
for i=1:size(I_seuil_propre,1) %verticalement
   for j=1:size(I_seuil_propre,2) %horizontalement
       if(I_seuil_propre(i,j)==1)
           sinusoide=j*cos(theta) + i*sin(theta);
           for t=1:1:length(sinusoide)
               H(ceil(sinusoide(t))+Ro_max,t) = 1 + H(ceil(sinusoide(t))+Ro_max,t);
           end
       end
   end
end

figure(5);
imagesc(H);
```

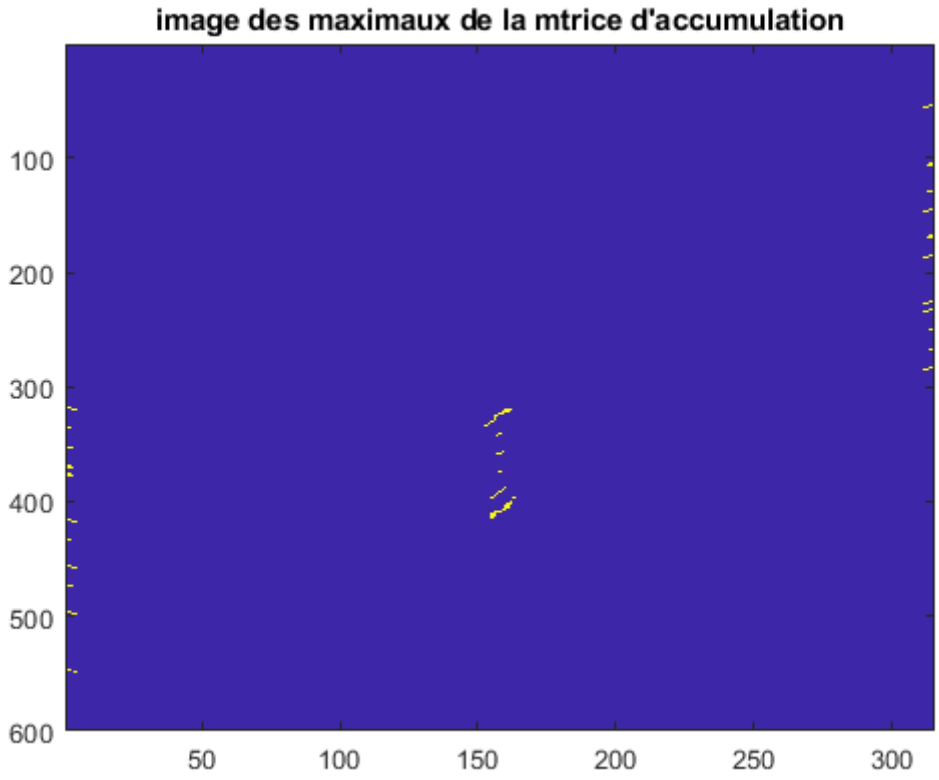
```
title('image de la matrice d''accumulation');
```



En effet cette matrice a été créée en prenant chaque pixel x,y puis en traçant dans le repère (Ro, thêta) la sinusoïde x*cos(thêta)+y*sin(thêta) en ajoutant 1 d'intensité à la matrice initialement nulle. De plus dans les faits nous avons ajouté Ro_max afin de pas avoir de Ro négatifs.

Dans ce repère rho thêta on remarque qu'il y a des Maximas locaux qui sont représentatif de plusieurs points pour un même thêta et rho ce qui caractérise une droite en coordonnés polaires. En effet il faut donc repérer ces maximas locaux afin de recréer les droites.

```
H_seuil = imbinarize(H,80);
figure(6);
imagesc(H_seuil);
title('image des maximaux de la mtrice d''accumulation');
pause(1);
```



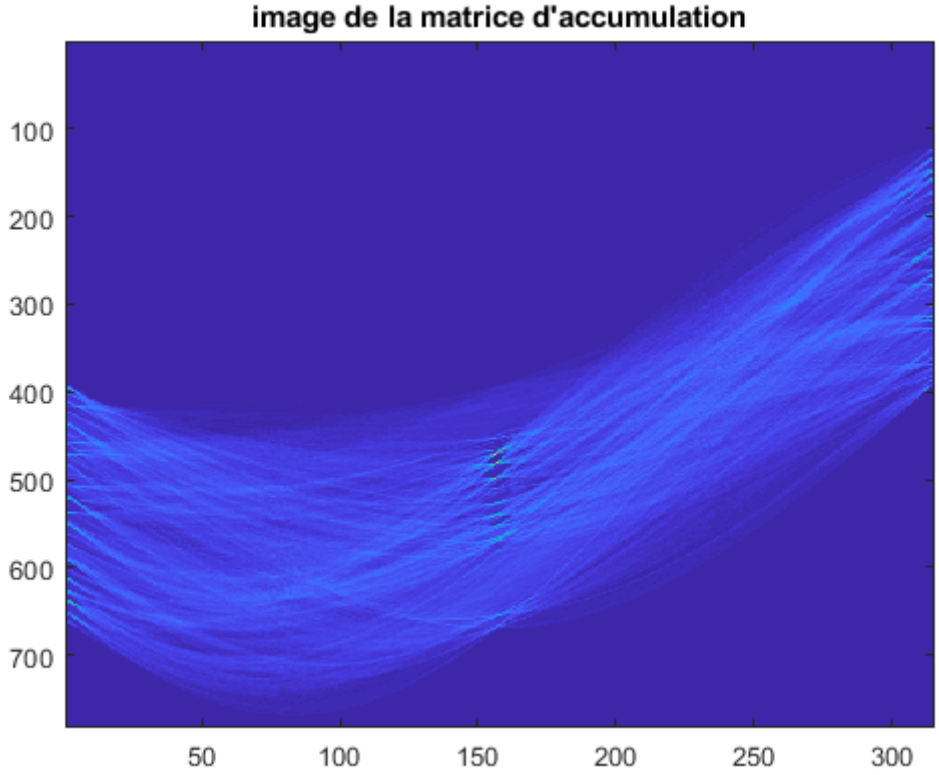
Les maximaux sont situés à des angles thêta de 0, pi/2,pi se qui correspond bien à des droites horizontales et verticales qui sont exclusivement présente sur l'image d'origine.

```
figure(7);
pause(1);
imshow(I);
hold on ;


theta=0:dtheta:pi-dtheta;
I_recons=zeros(size(I,1)+1,size(I,2)+1);
nb_droites=0;

for ro_h=1:size(H_seuil,1)
   for theta_h=1:size(H_seuil,2)
       if(H_seuil(ro_h,theta_h)==1)
           %construire droite de paramÃ¨tre (ro_h,theta_h) dans le plan xy
           x=1:272;
           vrai_theta=theta_h/100;
           plot(x,((ro_h-Ro_max)-x*cos(vrai_theta))/sin(vrai_theta));%tracÃ© de la
droite
           axis([0 size(I,2) 0 size(I,1)]);
        end
    end
end
```

```
title('image d''origine avec les lignes reconstruites');
```

On remarque bien que l'algorithme affiche des lignes à chaque droite sur les contours des lettres de « HELLO » ce qui est le comportement attendu.

On peut ensuite afficher les résultats pour l'image du circuit :



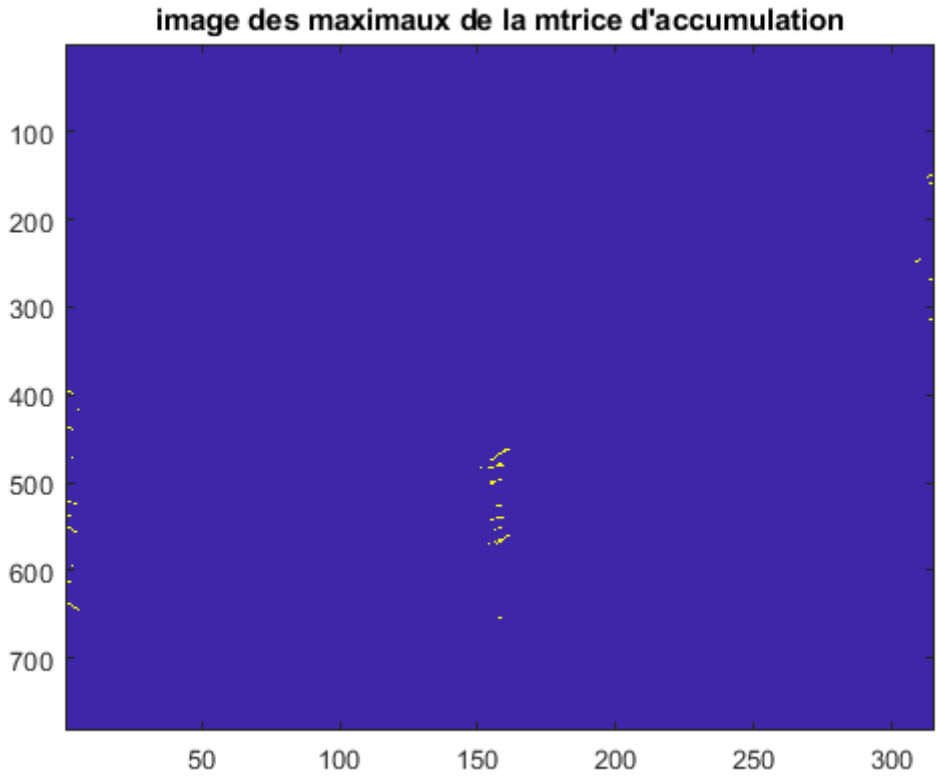


image d'origine avec les lignes reconstruites

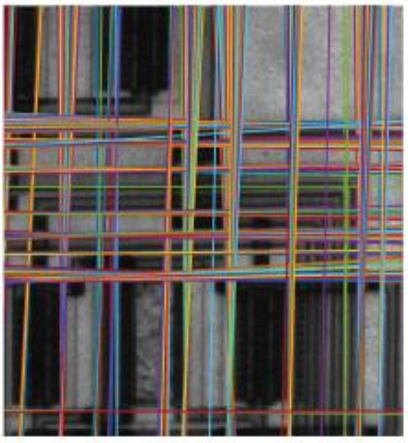


image du gradient avec les lignes reconstruites

On remarque les lignes reconstruite correspondent parfaitement aux lignes présentent sur les contours de l'image, pour améliorer le résultat de la reconstruction des lignes il faudrait améliorer la qualité des contours.

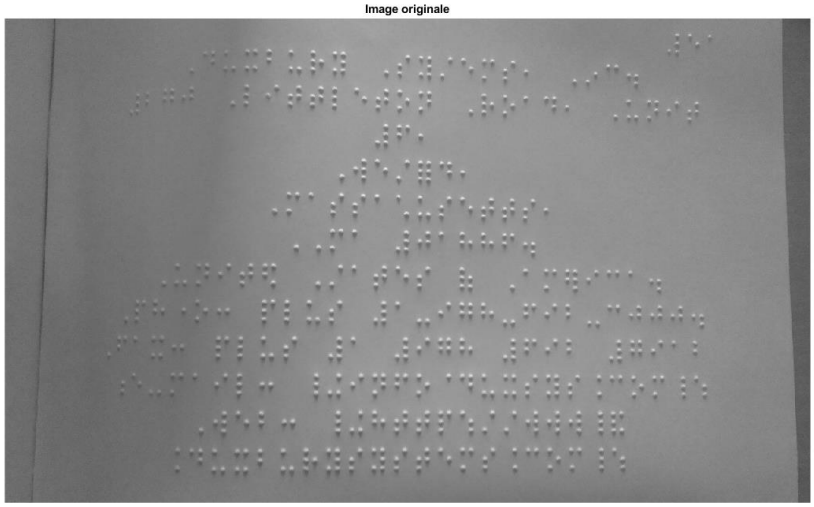
Cette méthode est plutôt efficace pour pouvoir repérer les ligne d'une image cependant il nécessite au préalable un travaille sur l'image afin avoir seulement les contours

II) Detection de cercles

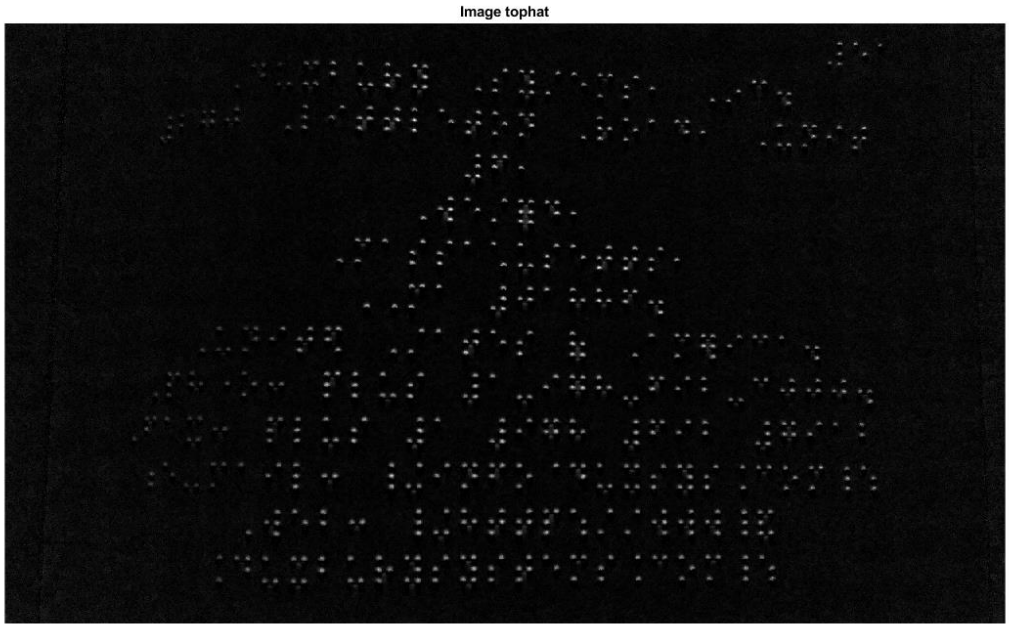
1) Obtention du gradient de l'image

Tout comme précédemment nous devons obtenir une image seuillé des cercles à détecter pour avoir de bons résultats.

```
Image2 = rgb2gray(im2double(imread('braille1.png')));
figure(1)
imshow(Image2)
title('Image originale')
Struct1 = strel('disk',10);
TopHat = imtophat(Image2,Struct1);


figure(2)
imshow(TopHat,[])
title('Image tophat')


Seuil1 = imbinarize(TopHat,0.1);

figure(3)
imshow(Seuil1,[])
title('Image tophat seuillé')
```



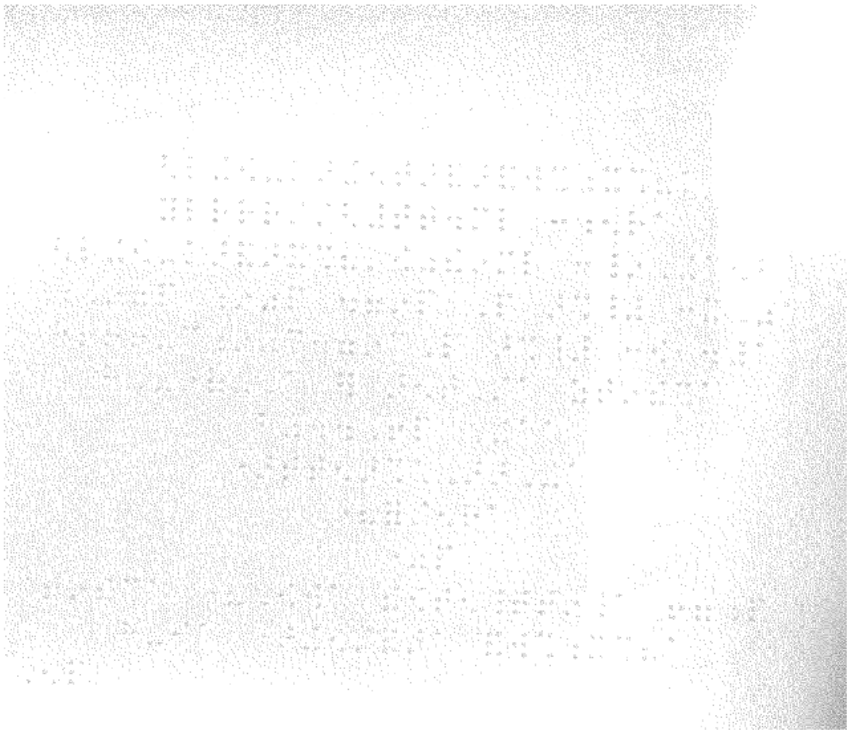
Pour cette image d'origine nous réalisons un tophat pour améliorer le contraste de l'image ainsi qu'un seuil de l'image obtenue :



Pour l'image2 on a :

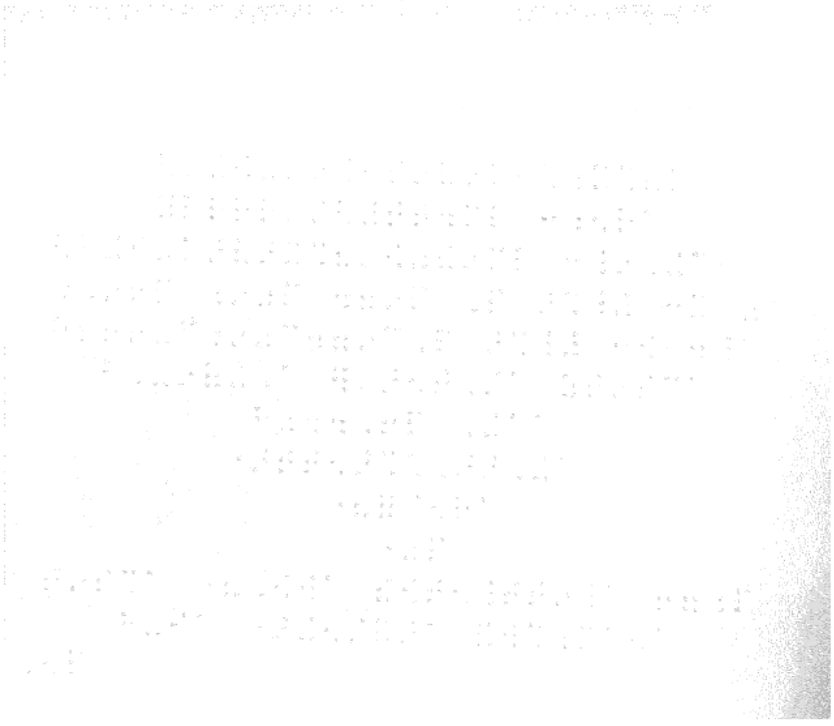
```
Image3 = rgb2gray(im2double(imread('braille2.png')));

figure(4)
imshow(Image3)
title('Image originale')

Struct1 = strel('rectangle',[1,2]);
Dil1 = imdilate(Image3, Struct1);
figure(5)
imshow(Dil1)
title('Image dilaté')

Struct2 = strel('disk',2);
BottomHat = imbothat(Dil1,Struct2);
Seuil3 = imbinarize(BottomHat,0.05);
figure(6)
imshow(Seuil3,[])
title('Image botttomhat seuillé')

Struct3 = strel('disk',3);
Seuil3 = imclose(Seuil3, Struct3);
Seuil3 = imclose(Seuil3, Struct3);
Seuil3 = imclose(Seuil3, Struct3);
figure(7)
imshow(Seuil3,[])
title('ouverture Image Bottomhat seuillé ')
```

Image originale

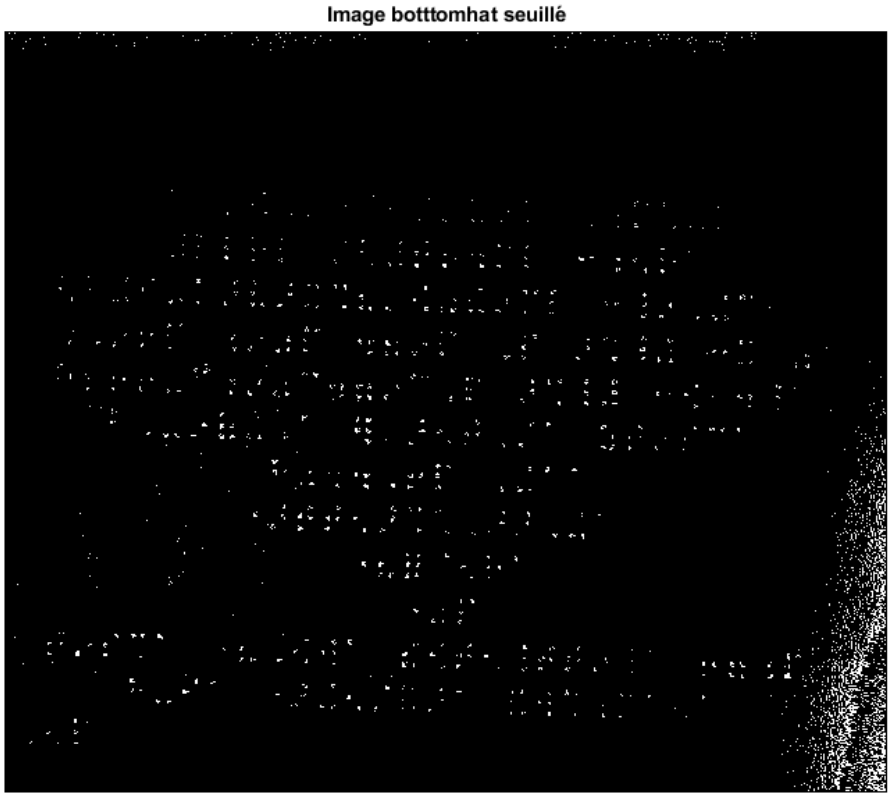


L’image étant très bruité l’algorithme de détection de cercle serait faussé, pour cela nous réalisons une dilatation :

Image dilaté



Nous réalisons ensuite des ouverture et un seuil pour tenté de rendre les cercles rond.

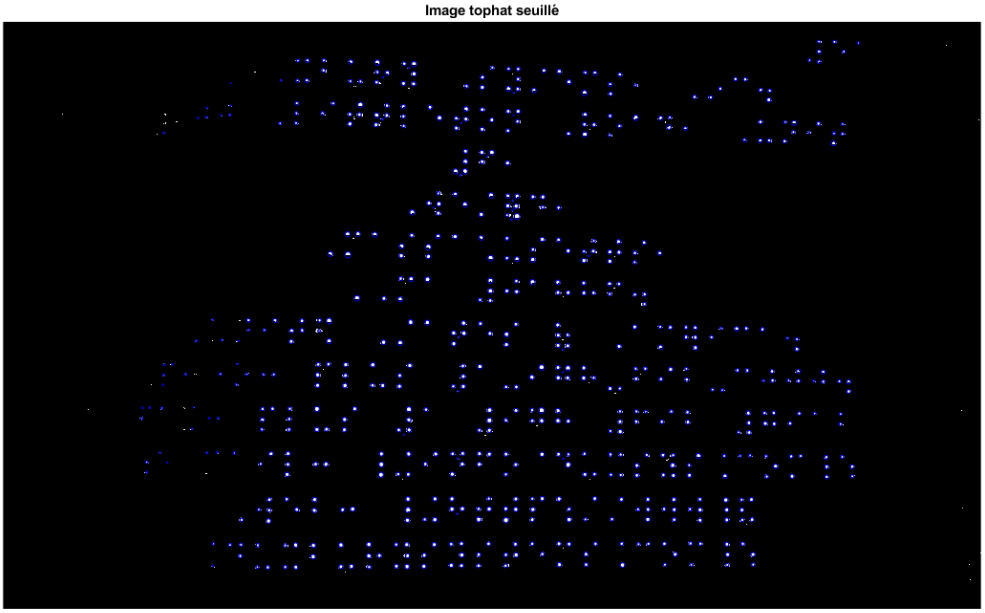


2) Méthode de Hough

Nous appliquons ensuite la méthode de Hough généralisé à la détection de cercle ce qui nous permet d'obtenir les résultats suivants :

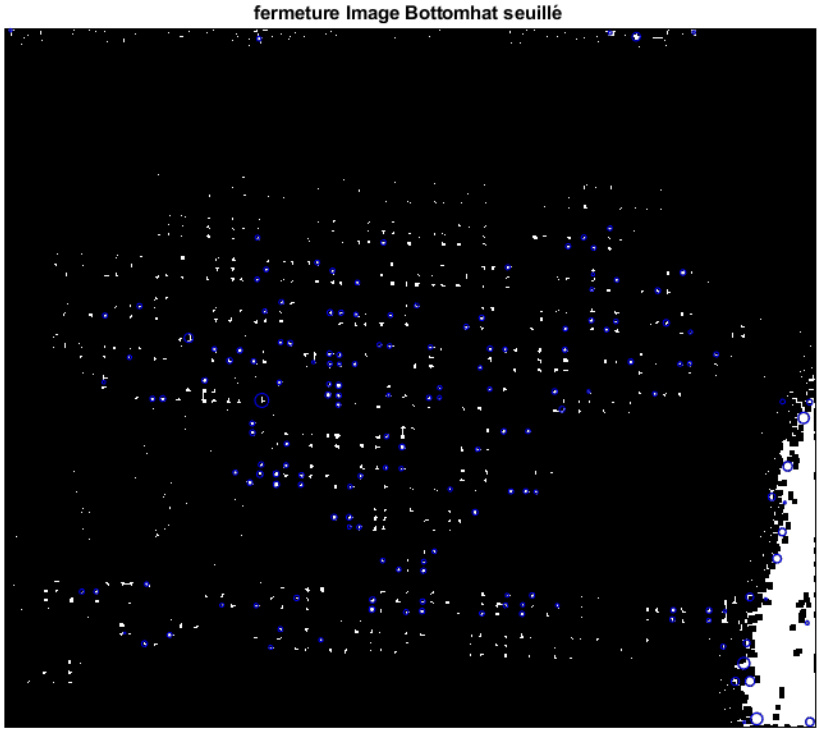
```
%permet de detecter les cercle de rayans entre 1 et 10
Rmin = 1;
Rmax = 10;
[centersBright, radiiBright] = imfindcircles(Seuil1,[Rmin Rmax]);
viscircles(centersBright, radiiBright, 'Color', 'b','LineWidth',0.1);
```

Image tophat seuillé



```
%permet de detecter les cercle de rayans entre 1 et 7
Rmin = 1;
Rmax = 7;
[centersBright, radiiBright] = imfindcircles(Seuil3,[Rmin Rmax]);
viscircles(centersBright, radiiBright, 'Color', 'b','LineWidth',0.1);
```

fermeture Image Bottomhat seuillé



La méthode semble plutôt efficace pour détecter des cercles Mais tout comme précédemment cette méthode dépend d'un bon travail préalable pour l'image d'origine.

Nous n'avons pas vraiment réussie à avoir de bons résultats pou la deuxièmes image du fait de notre travail préalable non parfait.