

Traitement d'images TP4 : Détection de contour

Line POUVARET, Hamdi BENAOUN

2015-2016

2. Dérivation par simples différences finies

2.1 Filtre de Roberts

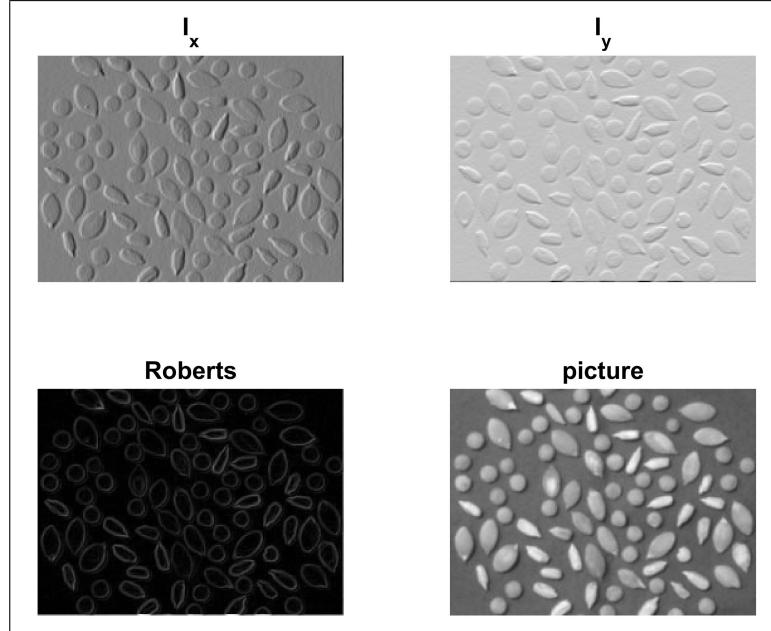
- Fonction roberts_differential, dans MatLab :

```
1 pic_x = conv2(pic, D, 'same');
2 pic_y = conv2(pic, D', 'same');
```

- calcul de la norme, dans MatLab :

```
1 pic_norm = sqrt(power(pic_x, 2)+power(pic_y, 2));
```

- Images des gradients en x, y et de la norme :



Le gradient en x prend des valeurs qui varient de -98 à 98.

Le gradient en y prend des valeurs qui varient de -232 à 73.

La norme prend des valeurs qui varient de 0 à 232.

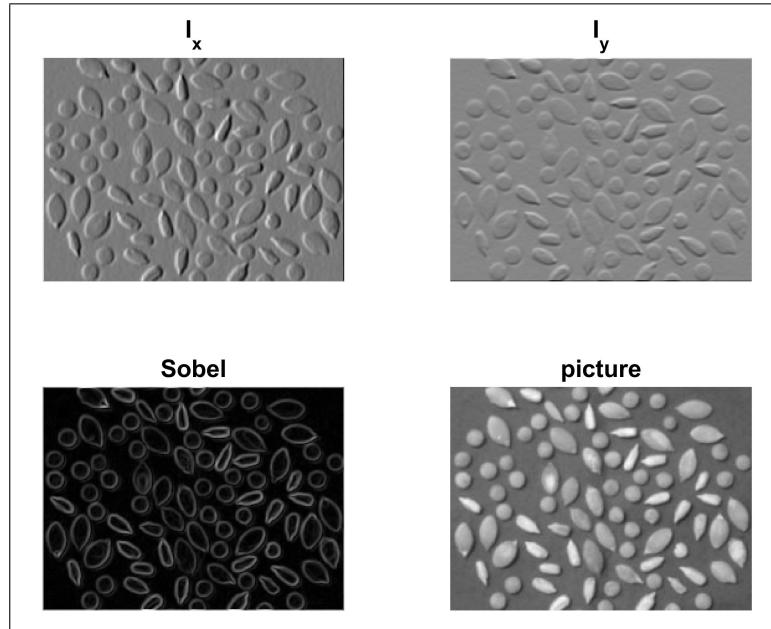
On remarque au niveau de l'image de la norme, que certaines ombres des graines sont détectées comme des contours (c'est logique puisqu'il y a une variation du sombre au clair).

2.2 Filtre de Sobel

- Fonction sobel_differential dans MatLab :

```
1 S=[1 ,2 ,1];
2 D=[1 ,0 , -1];
3 Gx=S'*D;
4 Gy=D'*S;
5
6 pic_x = conv2(pic, (1/4)*Gx, 'same');
7 pic_y = conv2(pic, (1/4)*Gy, 'same');
8
9 pic_norm = sqrt(power(pic_x, 2)+power(pic_y, 2));
```

- Images des gradients en x, y et de la norme :

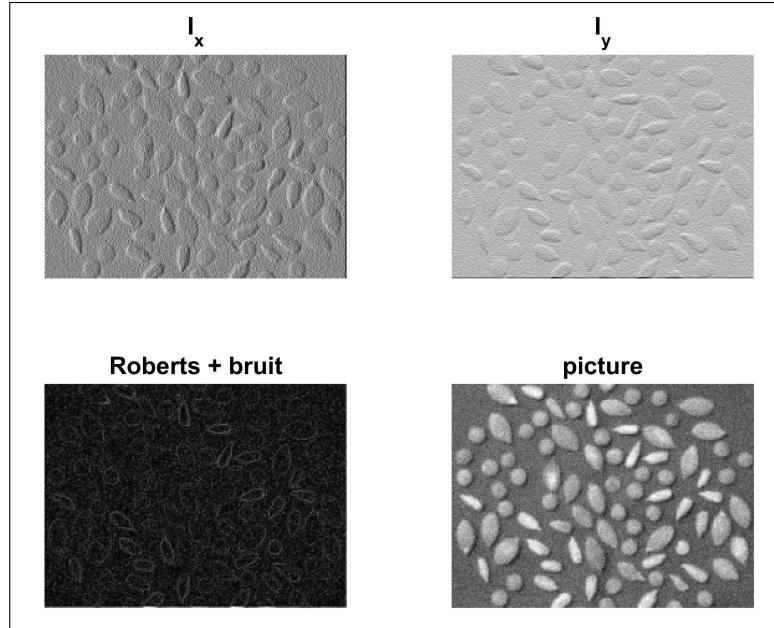


Les vrais contours des graines sont beaucoup plus nets. On constate tout de même encore un peu les ombres qui ont été détectées mais elles sont beaucoup moins visibles.

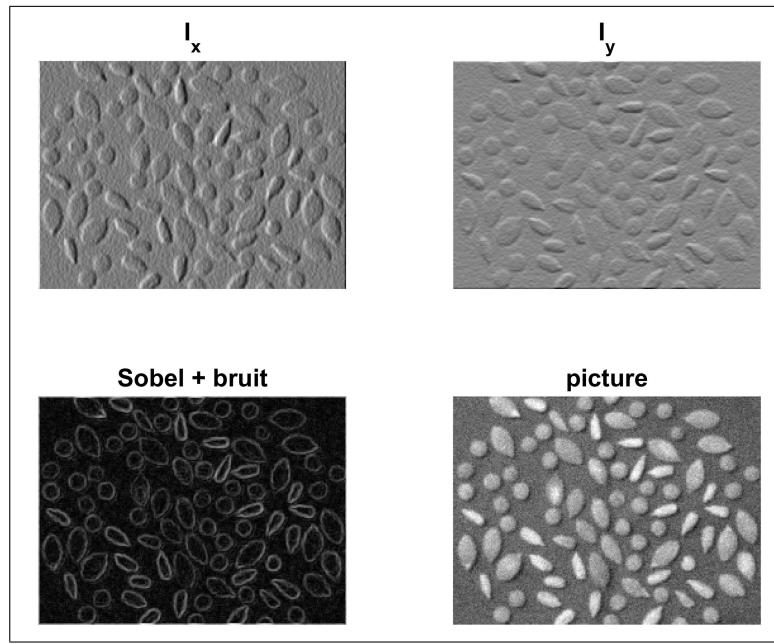
2.3 Ajout d'un bruit gaussien

On ajoute un bruit gaussien avec RSB = 10db.

- Avec le filtre de Roberts :



- Avec le filtre de Sobel :



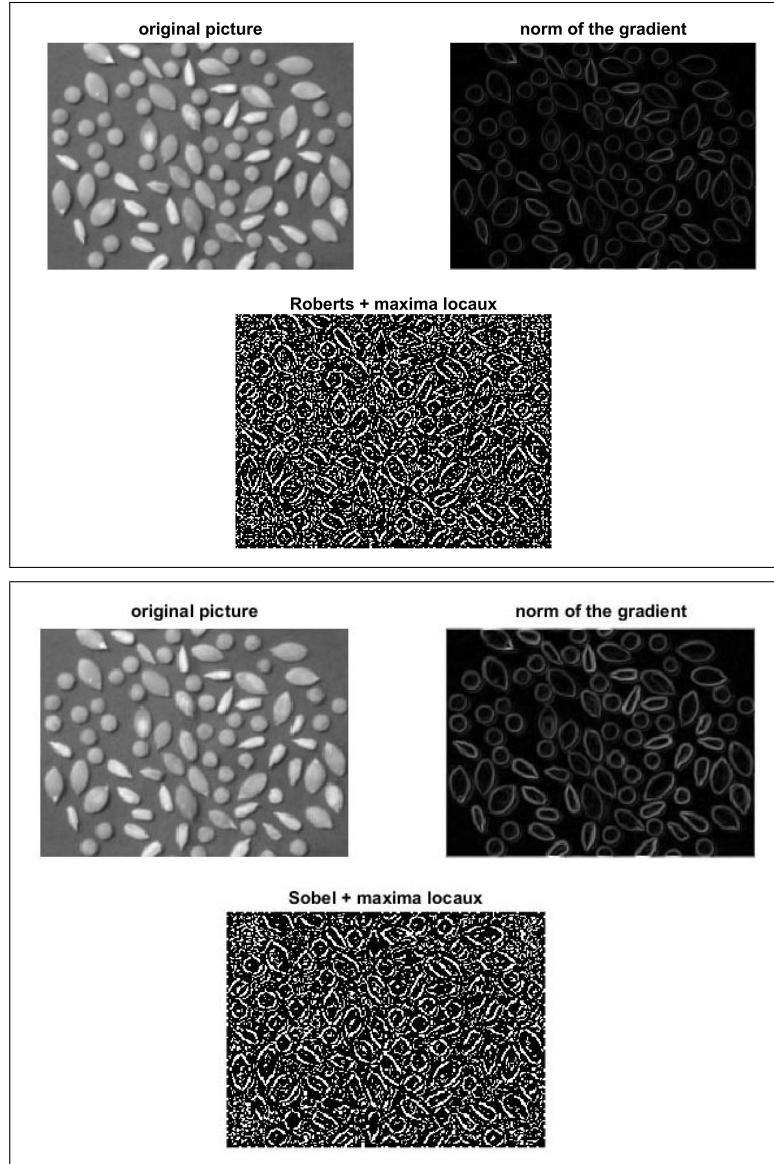
On constate que la méthode utilisant le filtre de Roberts n'est vraiment pas efficace dès qu'on a du bruit sur l'image.

En effet, on a du mal à constater correctement les contours des graines.

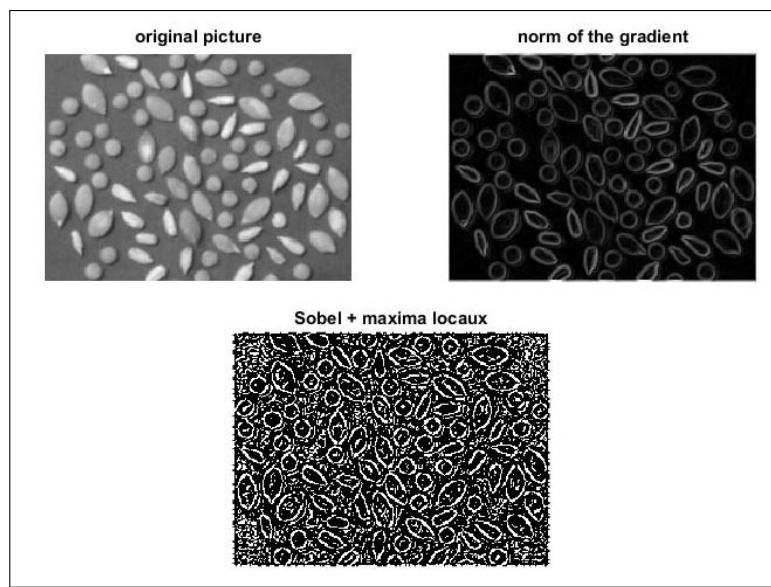
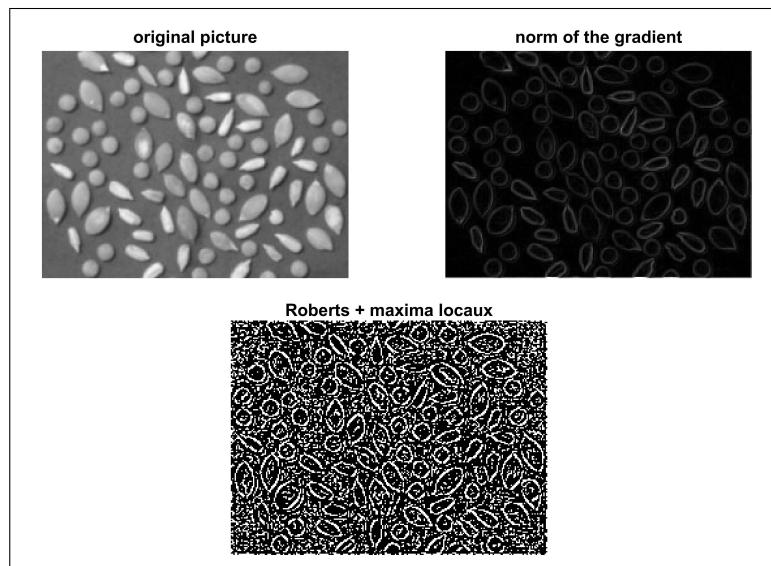
La méthode utilisant le filtre de Sobel est beaucoup plus robuste face au bruit car les contours des graines sont assez bien conservés même si on observe légèrement les contours correspondant au bruit dans l'image de la norme.

3. Recherche des maxima locaux dans la direction du gradient

- Interpolation au plus proche voisin (nearest)



- Interpolation bilinéaire



Sobel est encore une fois plus efficace pour rechercher les maxima locaux (en nearest ou bilinéaire).
En bilinéaire, les contours sont globalement plus nets (pour Sobel ou Roberts).

4. Seuillage et chaînage des points de contour

- Fonction "link_picture" dans MatLab :

```

1 %Passe No 1
2 for i = 2:N_i-1
3     for j = 2:N_j-1
4         if is_edges_points(i,j)==1
5             v1 = [i, j-1];
6             v2 = [i-1, j-1];
7             v3 = [i-1, j];
8             v4 = [i-1, j+1];
9             if edges_points(v1(1,1), v1(1,2)) == 1

```

```

10         edges_points(i,j) = 1;
11     end
12     if edges_points(v2(1,1), v2(1,2)) == 1
13         edges_points(i,j) = 1;
14     end
15     if edges_points(v3(1,1), v3(1,2)) == 1
16         edges_points(i,j) = 1;
17     end
18     if v4(1,1) <= 180 && v4(1,1) > 0 && v4(1,2) > 0 && v4(1,2) <= 243
19         if edges_points(v4(1,1), v4(1,2)) == 1
20             edges_points(i,j) = 1;
21         end
22     end
23 end
24 end
25
26
27 %Passe No 2
28 for j = N_j-1:-1:2
29     for i=2 :N_i-1
30         if is_edges_points(i,j)==1
31             v1 = [i-1, j];
32             v2 = [i-1, j+1];
33             v3 = [i, j+1];
34             v4 = [i+1, j+1];
35             if edges_points(v1(1,1), v1(1,2)) == 1
36                 edges_points(i,j) = 1;
37             end
38
39             if edges_points(v2(1,1), v2(1,2)) == 1
40                 edges_points(i,j) = 1;
41             end
42
43             if edges_points(v3(1,1), v3(1,2)) == 1
44                 edges_points(i,j) = 1;
45             end
46
47             if v4(1,1) <= 180 && v4(1,1) > 0 && v4(1,2) > 0 && v4(1,2) <= 243
48                 if edges_points(v4(1,1), v4(1,2)) == 1
49                     edges_points(i, j) = 1;
50                 end
51             end
52         end
53     end
54 end
55
56 %Passe No 3
57 for i =N_i-1:-1:2
58     for j=N_j-1:-1 :2
59         if is_edges_points(i,j)==1
60             v1 = [i, j+1];
61             v2 = [i+1, j+1];
62             v3 = [i+1, j];
63             v4 = [i+1, j-1];
64
65             if edges_points(v1(1,1), v1(1,2)) == 1
66                 edges_points(i,j) = 1;
67             end
68

```

```

69         if edges_points(v2(1,1), v2(1,2)) == 1
70             edges_points(i,j) = 1;
71         end
72
73         if edges_points(v3(1,1), v3(1,2)) == 1
74             edges_points(i,j) = 1;
75         end
76
77         if v4(1,1) <= 180 && v4(1,1) > 0 && v4(1,2) > 0 && v4(1,2) <= 243
78             if edges_points(v4(1,1), v4(1,2)) == 1
79                 edges_points(i, j) = 1;
80             end
81         end
82     end
83 end
84
85 %Passe No 4
86 for j =2 :N_j-1
87     for i=N_i-1:-1 :2
88         if is_edges_points(i,j)==1
89             v1 = [i+1, j];
90             v2 = [i+1, j-1];
91             v3 = [i, j-1];
92             v4 = [i-1, j-1];
93
94             if edges_points(v1(1,1), v1(1,2)) == 1
95                 edges_points(i,j) = 1;
96             end
97
98             if edges_points(v2(1,1), v2(1,2)) == 1
99                 edges_points(i,j) = 1;
100            end
101
102            if edges_points(v3(1,1), v3(1,2)) == 1
103                edges_points(i,j) = 1;
104            end
105
106
107            if v4(1,1) <= 180 && v4(1,1) > 0 && v4(1,2) > 0 && v4(1,2) <= 243
108                if edges_points(v4(1,1), v4(1,2)) == 1
109                    edges_points(i, j) = 1;
110                end
111            end
112        end
113    end
114 end

```

5. Exécution de la chaîne complète de détection de contours

- Contours obtenus
 - Méthode de Roberts

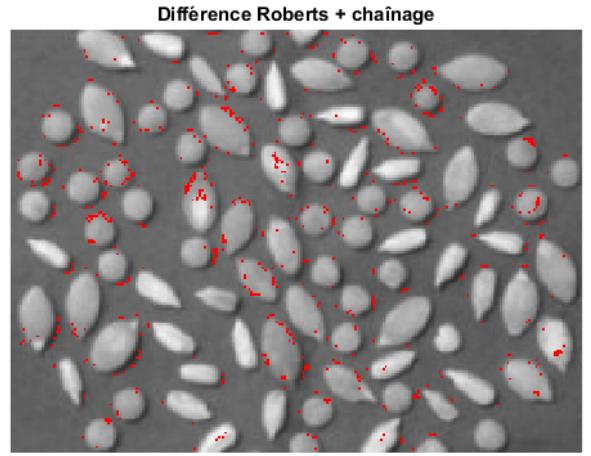
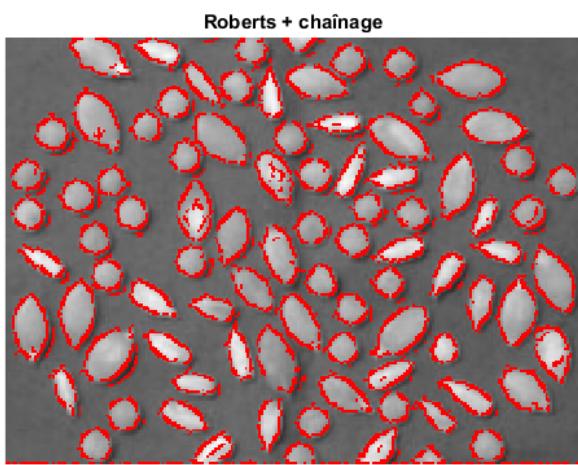
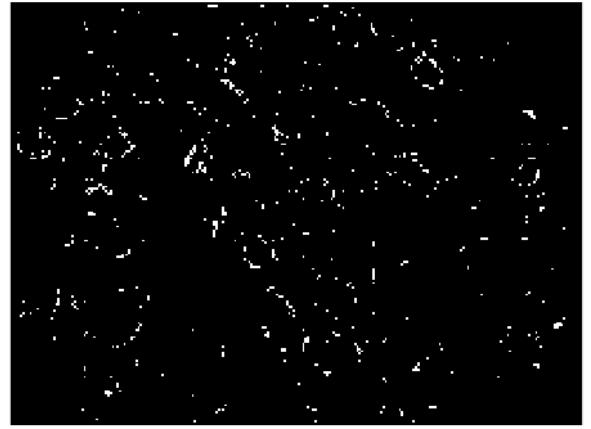
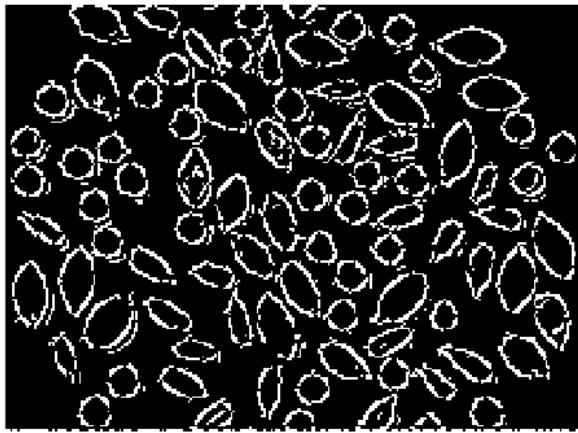


FIGURE 1 – Chaînage sans bruit

FIGURE 2 – Chaînes rajoutées (sans bruit)

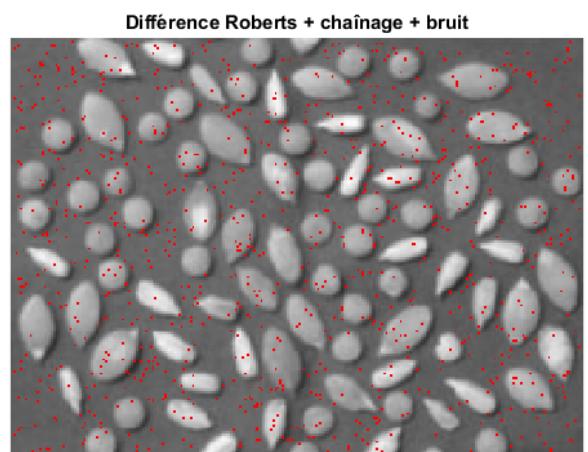
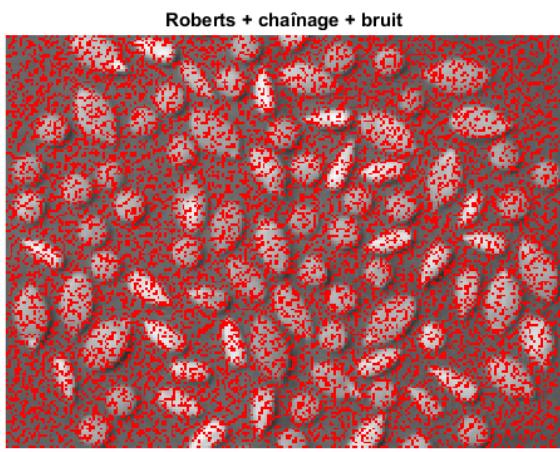
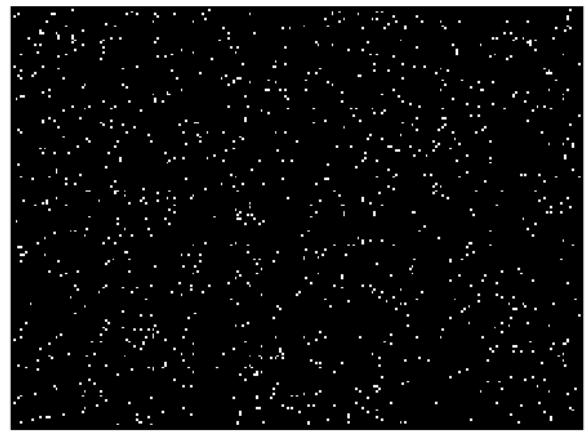
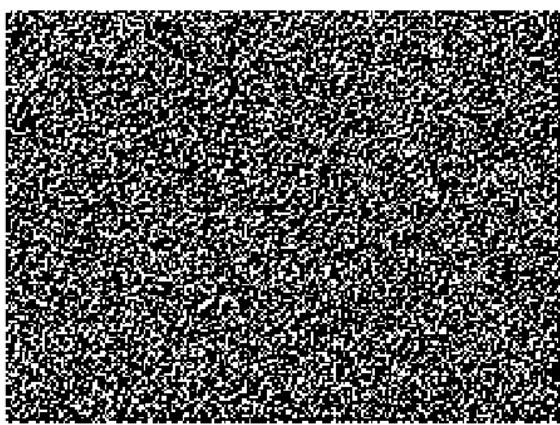


FIGURE 3 – Chaînage avec rsb=5db

FIGURE 4 – Chaînes rajoutées (rsb=5db)

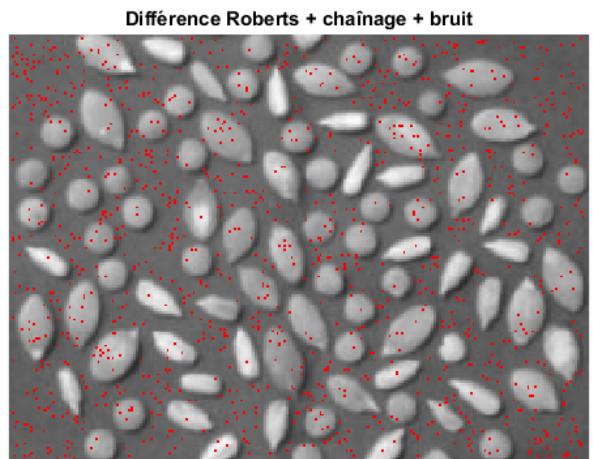
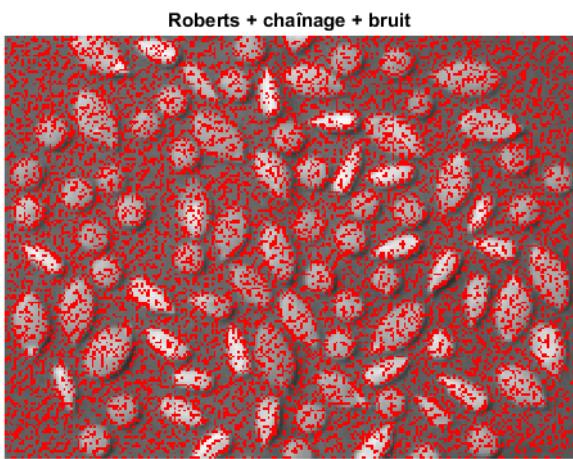
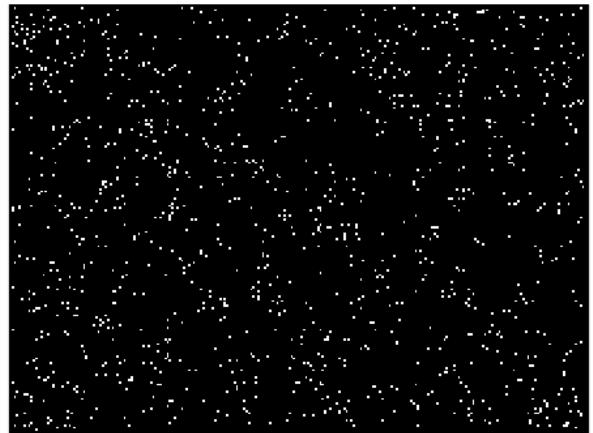
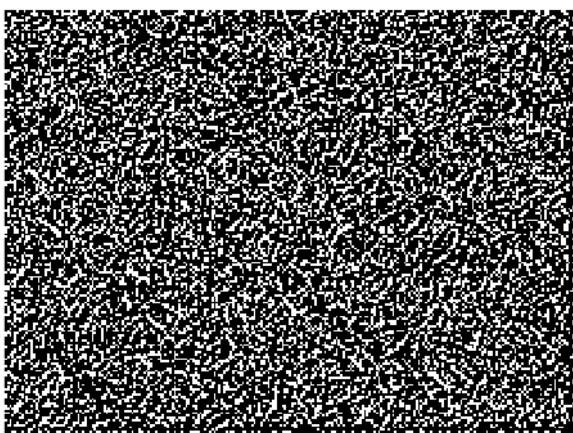


FIGURE 5 – Chaînage avec rsb=10db

FIGURE 6 – Chaînes rajoutées (rsb=10db)

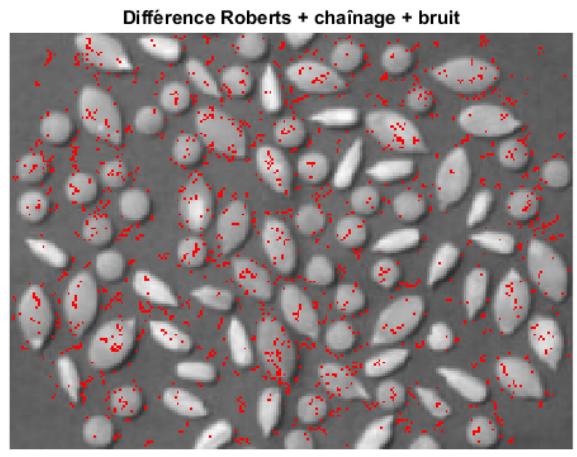
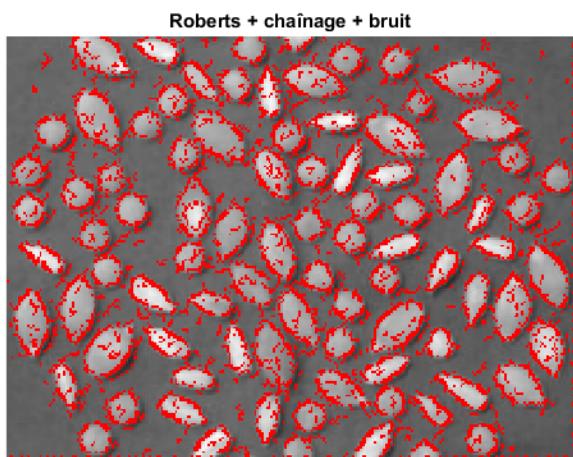
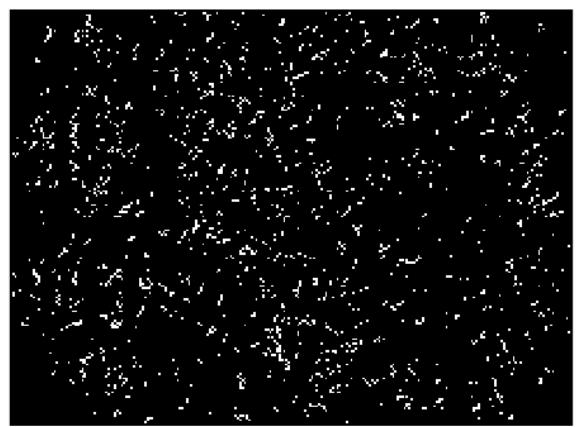
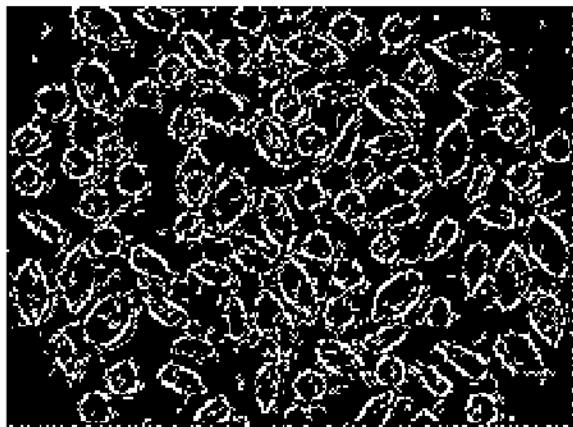


FIGURE 7 – Chaînage avec rsb=20db
– Méthode de Sobel

FIGURE 8 – Chaînes rajoutées (rsb=20db)

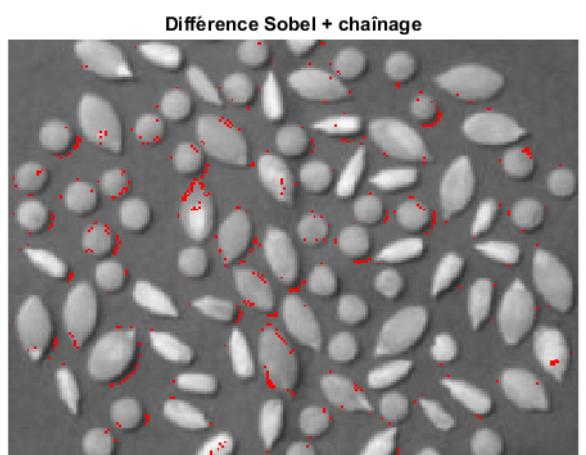
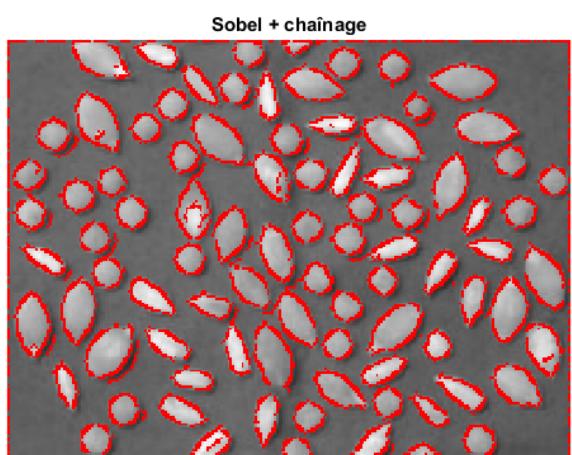
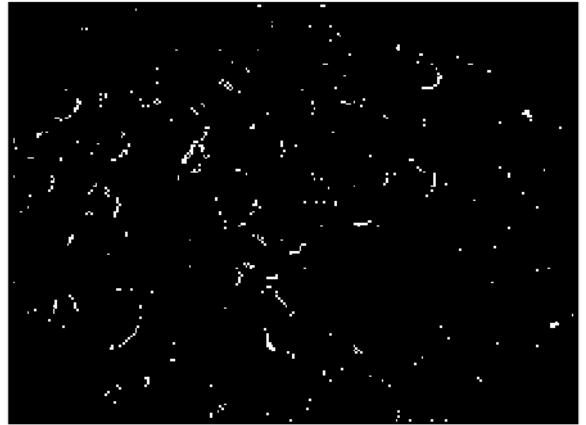
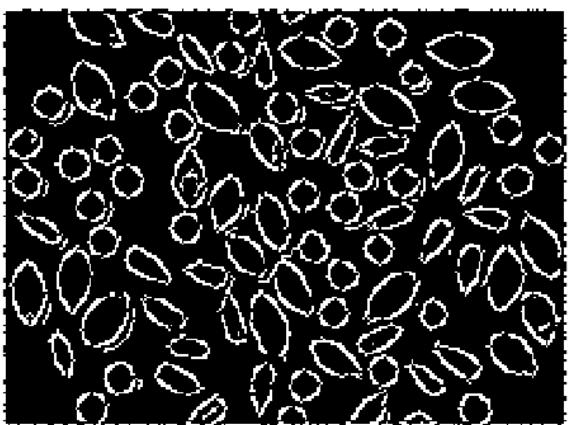


FIGURE 9 – Chaînage sans bruit

FIGURE 10 – Chaînes rajoutées (sans bruit)

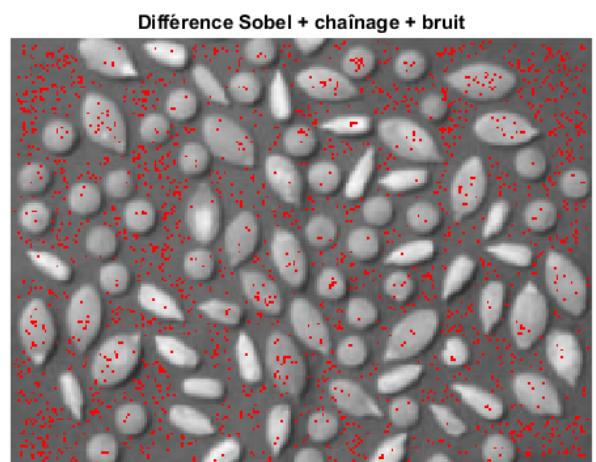
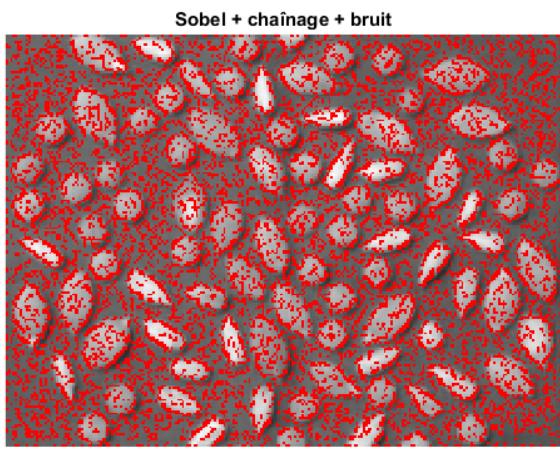
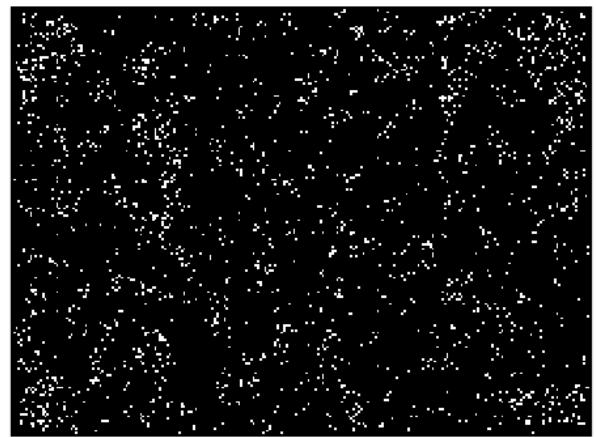
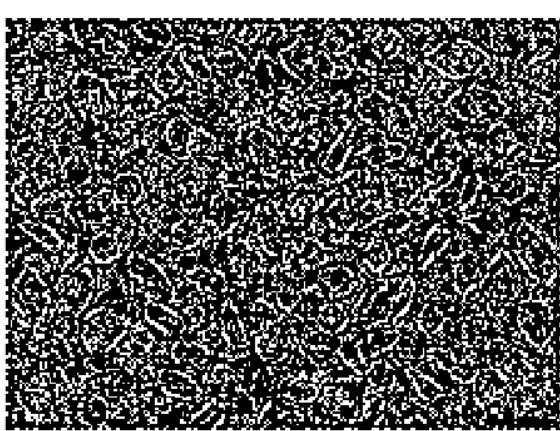
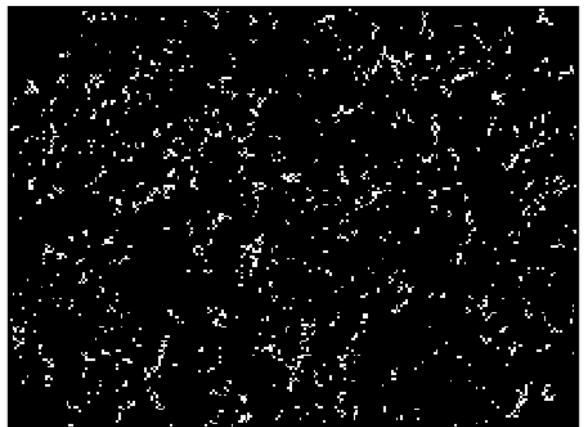
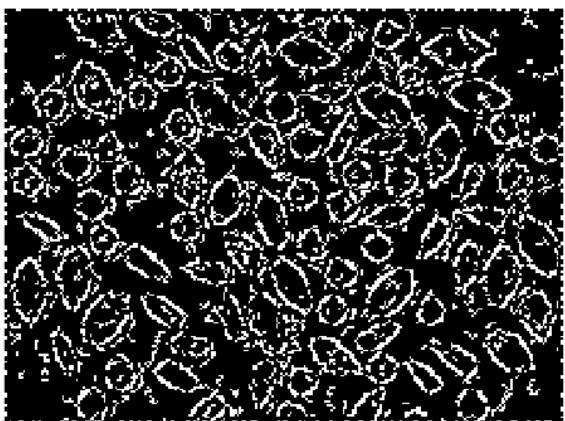
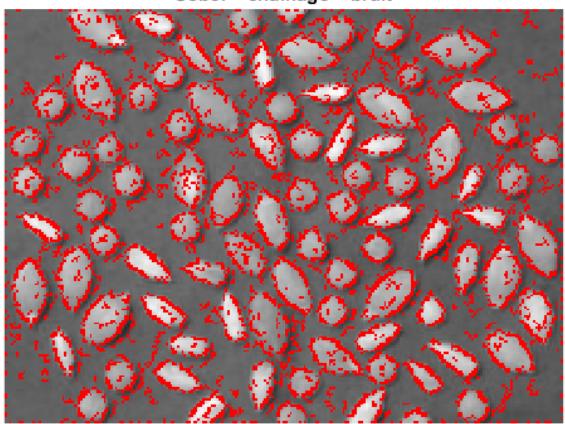


FIGURE 11 – Chaînage avec rsb=5db

FIGURE 12 – Chaînes rajoutées (rsb=5db)



Sobel + chaînage + bruit



Différence Sobel + chaînage + bruit

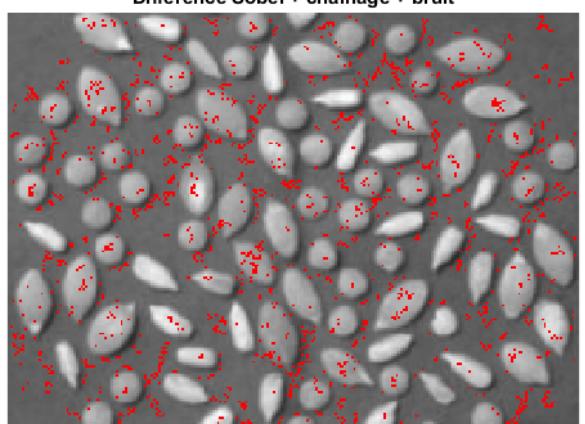


FIGURE 13 – Chaînage avec rsb=10db

FIGURE 14 – Chaînes rajoutées (rsb=10db)

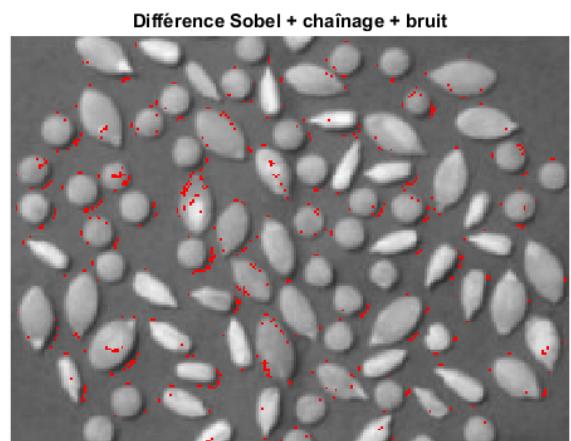
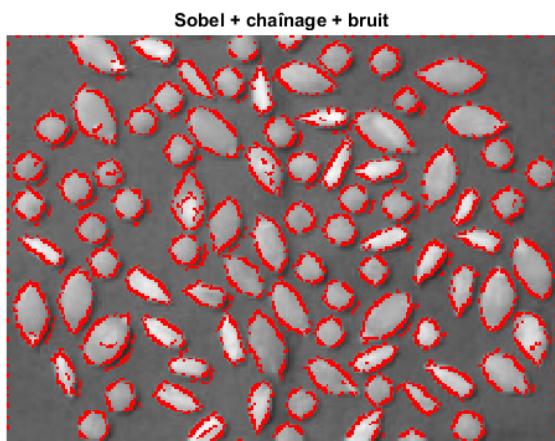
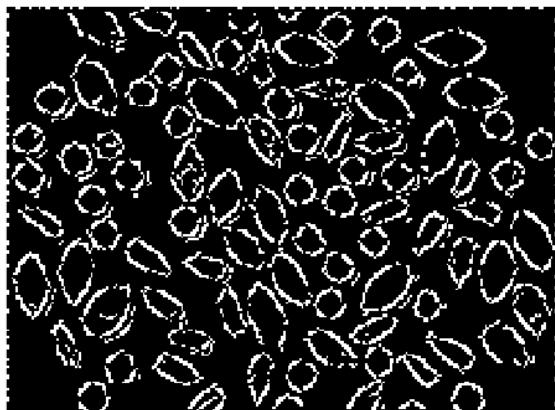


FIGURE 15 – Chaînage avec rsb=20db

FIGURE 16 – Chaînes rajoutées (rsb=20db)

- Lorsque l'on fait varier la valeur de d , on remarque un changement au niveau des contours.

En effet, lorsqu'on augmente la valeur de d , les contours détectés vont être plus épais mais moins contigus.

Si on rapproche la valeur de d de 1, les contours détectés sont donc plus fins mais parfois pas complets (on observe des trous dans les contours des graines).

Au niveau du bruit, on constate toujours que la méthode de Sobel est plus robuste même si à partir d'un rsb de 5db la détection des contours est beaucoup moins efficace.