BILLY Nicolas TP3 IMA

1 Contour

1.1 Filtre de gradient local par masque

- L'avantage du filtre pixel est la présence de facteurs plus importants sur la ligne du milieu par rapport aux lignes extérieurs, ceci va permettre d'effectuer une moyenne autour du pixel et ne va pas prendre en compte le bruit pour détecter des contours.
- Ainsi, il n'est pas nécessaire d'appliquer un filtre passe-bas avant d'utilsier ce filtre.

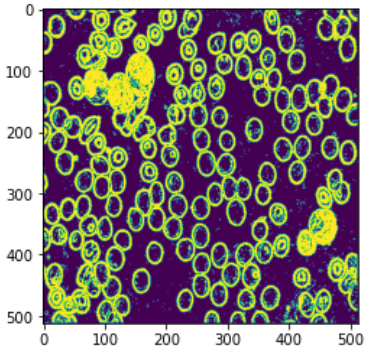

Figure 1: Gradient seuillé à 0.05

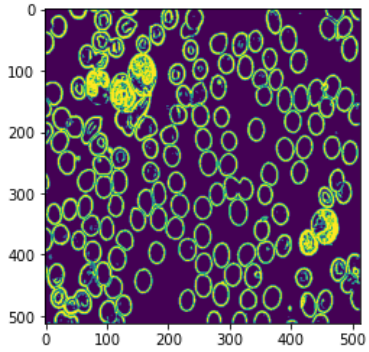

Figure 2: Gradient seuillé à 0.1

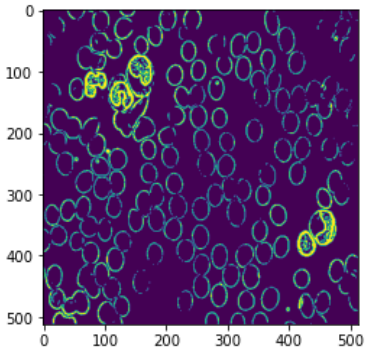

Figure 3: Gradient seuillé à 0.2

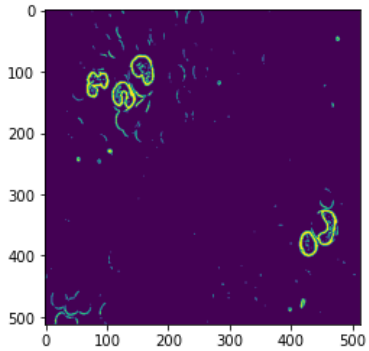

Figure 4: Gradient seuillé à 0.3

Plus on on augmente le seuil, moins les contours sont détectés, à partir d'un seuil à 0.3, une grande partie des contours ne sont plus détectés. A l'inverse, en diminuant ce seuil, trop de contours sont détectés, qui ne devraient pas forcément l'être dans certains cas d'étude.

1.2 Maximum du gradient filtré dans la direction du gradient

- Avec le procédé maximaDirectionGradient, il faut appliquer un filtre passe bas plus important car Sobel n'est pas suffisant pour enlever le bruit. Il semble que l'image soit plus homogène et que les contours soient continus et tous de même largeurs.

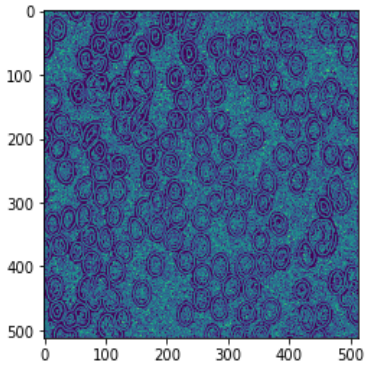

Figure 5: Maximum du gradient dans la direction du gradient, filtré avec $\sigma = 0.2$ et sans seuil

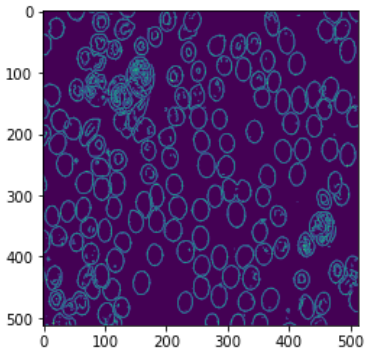

Figure 6: Gradient dans la direction maximale seuillé à 0.05

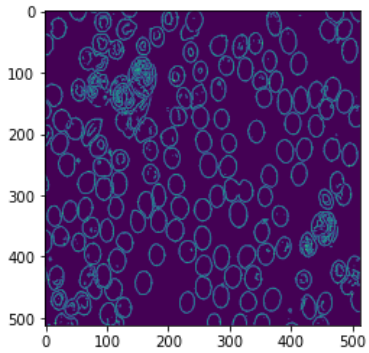

Figure 7: Gradient sdans la direction maximale euillé à 0.1

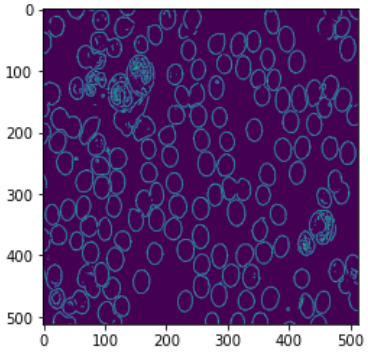

Figure 8: Gradient dans la direction maximale seuillé à 0.15

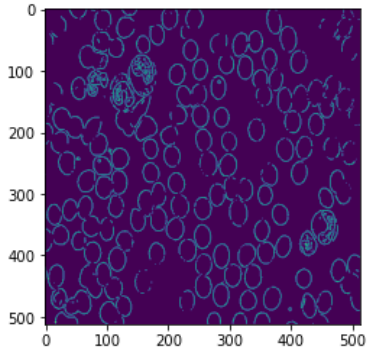

Figure 9: Gradient dans la direction maximale euillé à 0.2

- Plus le seuil est élévé, plus il y a de la robustesse au bruit.
- Pour un seuil à 0.15, il semble que les contours soient encore continus et que les cellules soient bien séparées. Il y a peu de cellules qui ont encore un peu de bruit, et si des cellules se chevauchent, elles seront détectées comme une unique cellule. Mais c'est le compromis à trouver.

1.3 Filtre récursif de Deriche

- Le temps de calcul ne dépend pas de α car le nombre d'étapes dans les boucles dépend uniquement de la taille de l'image.

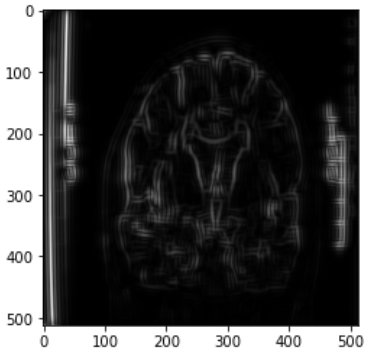

Figure 10: $\alpha = 0.3$

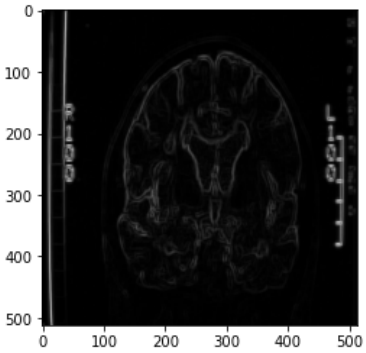

Figure 11: $\alpha = 1$

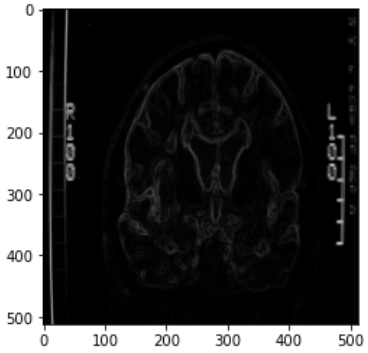

Figure 12: $\alpha = 2$

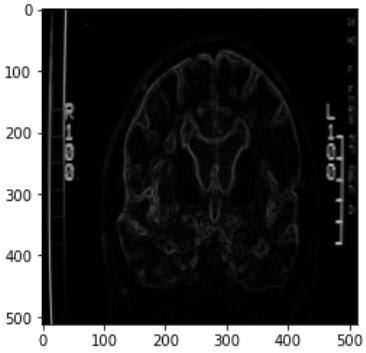

Figure 13: $\alpha = 3$

- Le but de DericheSmooth est de lisser les images encore plus. En effet dans le code de DerichGradx, on calcule gradx avec:

```
gradx[i,:]=c*ae*(b1-b2);
```

Alors qu'avec dericheSmooth on a:

```
smoothx[i,:]=b1+b2;
```

Donc on effectue une moyenne, ce qui va lisser l'image

1.4 Passage par zero du laplacien

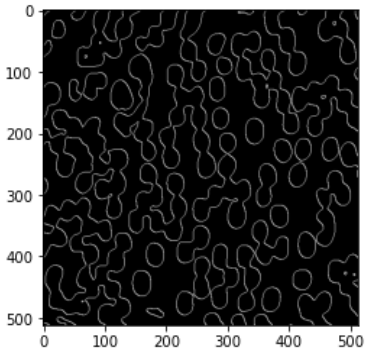

Figure 14: $\alpha = 0.3$

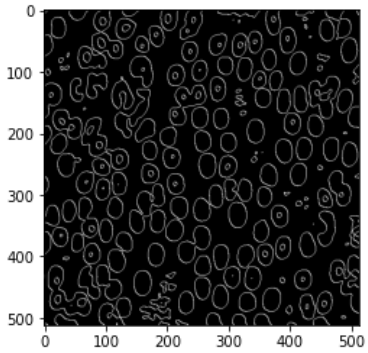

Figure 15: $\alpha = 0.5$

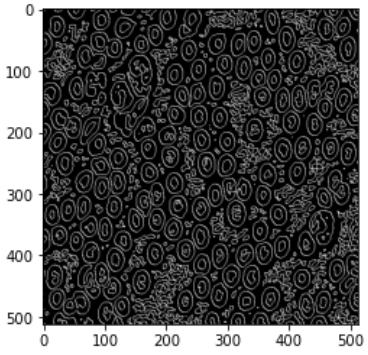

Figure 16: $\alpha = 1$

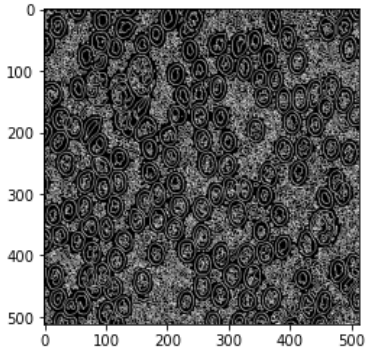

Figure 17: $\alpha = 2$

Il semble que plus α est élevé, plus on détecte de contours, et on voit plus de bruits dans l'image. A l'inverse, plus α est petit moins on détecte les contours et les effets du bruit sont réduits .
- Par rapport à la méthode de Deriche, les contours avec le Laplacien sont toujours fermés.
- Avec cette approche des faux contours apparaissent, pour résoudre ce problème, on peut introduire un seuil sur la norme des contours pour choisir seulement les plus importants.

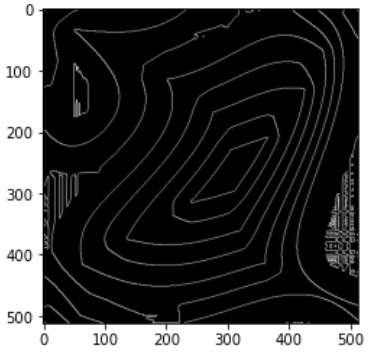

Figure 18: Laplacien, $\alpha = 1$, sans seuil

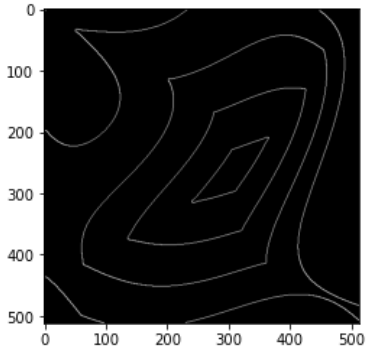

Figure 19: Norme du laplacien seuillé

1.5 Nouvelle image: pyramide

Dans cette image de pyramide avec bruit gaussien, la méthode la plus adaptée pour la segmentation est le passage par zéro du laplacien. Comme vu au dessus, cette technique permet de réaliser des contours fermés, hors ici on sait que sur l'image les contours sont fermés.

2 Seuillage par hysteresis

Si le rayon de l'élément structurant est pair, on détecte plus que pour un rayon impair voisin. Pour les rayons pairs, plus il est élevé moins tophat détecte les lignes, et pour les rayons impairs il semble que c'est l'inverse.

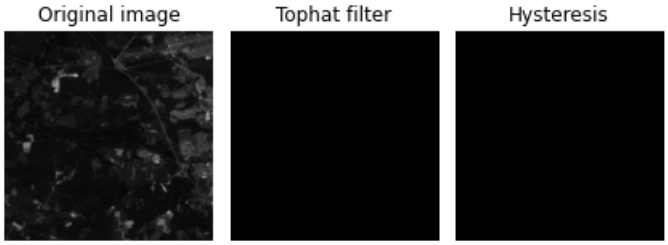

Figure 20: rayon 1

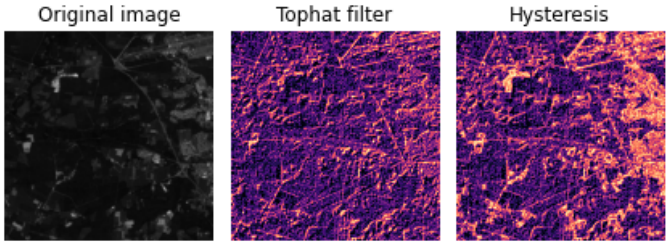

Figure 21: rayon 2

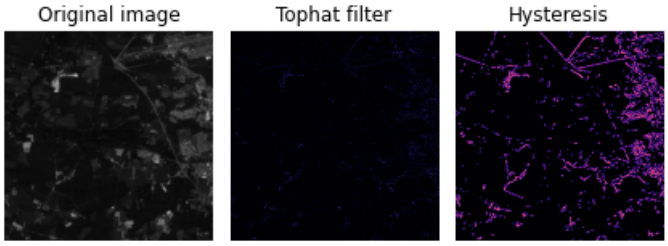

Figure 22: rayon 3

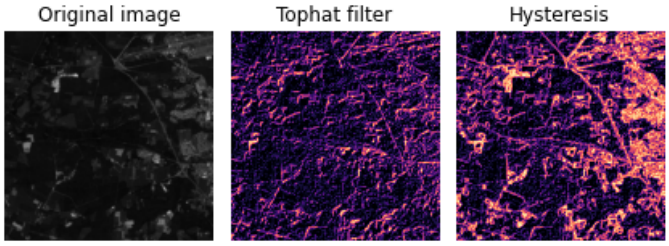

Figure 23: rayon 4

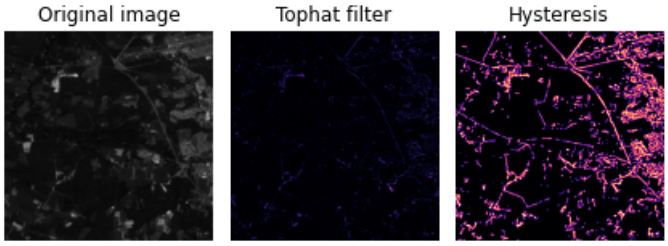

Figure 24: rayon 5

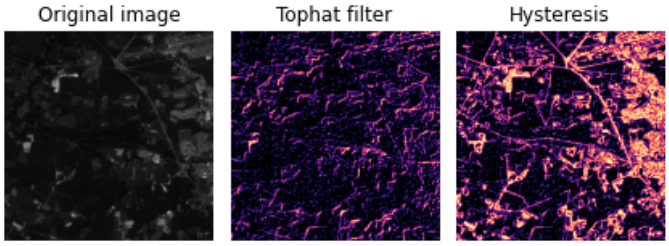

Figure 25: rayon 6

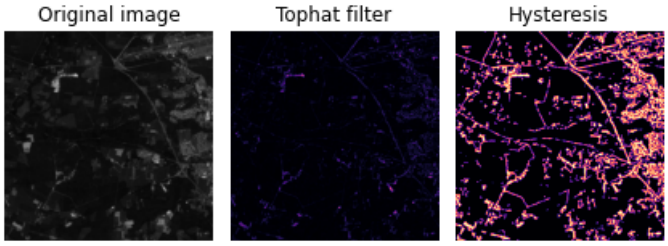

Figure 26: rayon 7

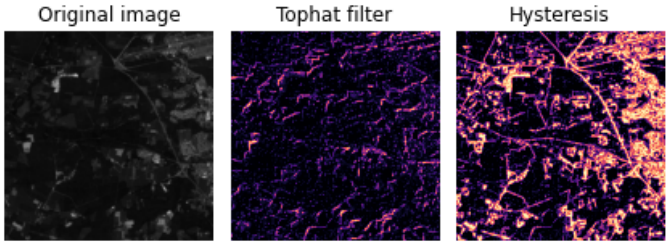

Figure 27: rayon 8

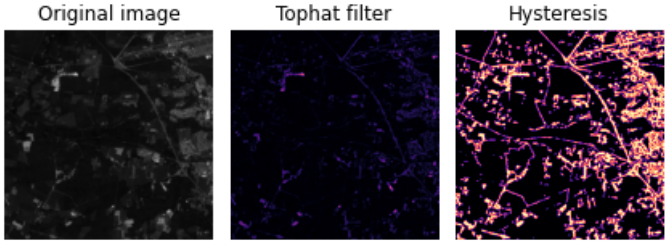

Figure 28: rayon 9

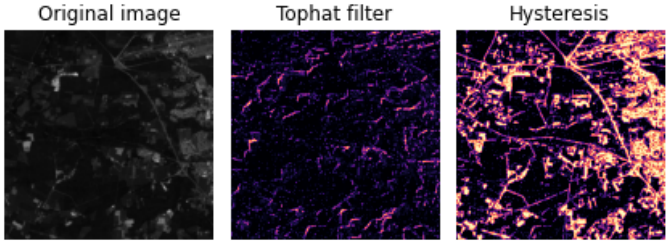

Figure 29: rayon 10

3 K-moyennes

3.1 Image à niveaux de gris

Figure 30: Algortihme des K-moyennes appliqué sur l'image cells avec 2 classses

- On a utilisé ci dessus l'initialisation kmeans++ qui choisit aléatoirement les centres de classes, on peut aussi donner une liste à l'initialisation qui contient les coordonnées des centres des classes.
- Avec cet algorithme, on voit que les centres des classes restent stables.
- Avec l'image muscletif, la difficulté rencontrée par l'algortihme K-means vient du fait que le centre des cellules musculaires est très clair.

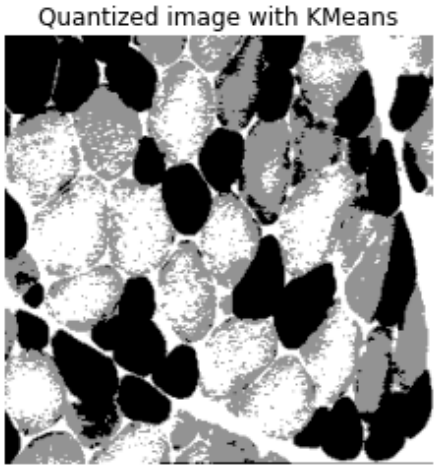

Figure 31: Algortihme des K-moyennes appliqué sur l'image muscle avec 3 classses

Le filtrage médian permet de lisser les zones intérieurs des cellules, et ainsi rendre plus homogène les centres claires, c'est pourquoi l'algorithme marchera mieux. Mais le résultat n'est toujours pas convaincant.

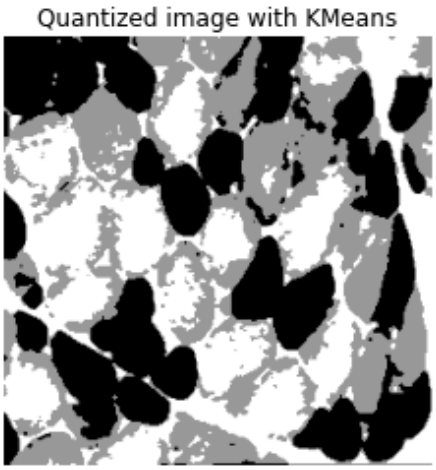

Figure 32: filtre médian taille 2

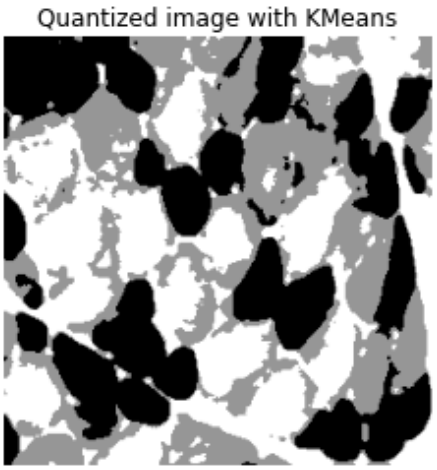

Figure 33: filtre médian taille 4

3.2 Image couleur

Figure 34: filtre médian taille 2

Figure 35: filtre médian taille 4

La dégradation de l'image se traduit par un aspect de l'image moins vif, la teinte est plus sombre. De plus on voit qu'il y a un lissage des textures.

Avec 12 classes, le résultat commence à être assez fidèle à l'image originale, seule les tiges vertes ne sont pas aussi vives.

Figure 36: fleurs avec 6 classes

Figure 37: fleurs avec 12 classes

Pour retrouver les planches, on peut effectuer un filtrage avant pour faire disparaitre les annotations de la carte IGN comme les rayures des différentes zones. On applique par exemple un filtre médian de taille 3.

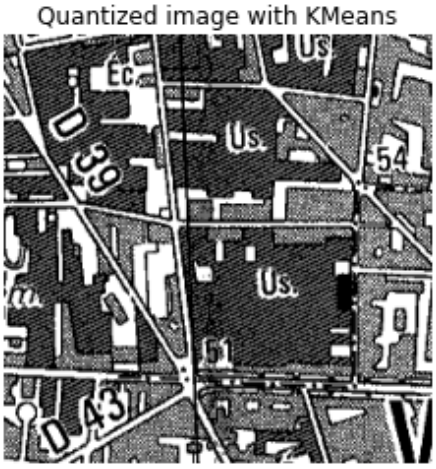

Figure 38: Kmeans avec 3 classes

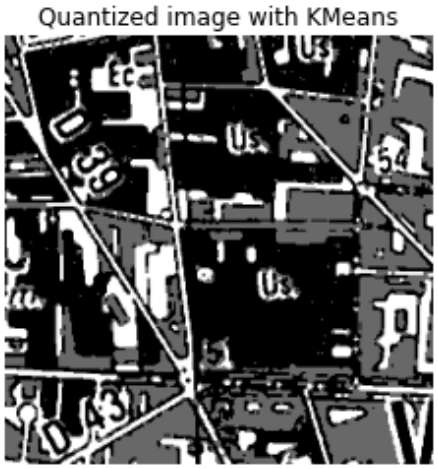

Figure 39: Kmeans appliqué à l'image filtré (median de taille 3)

4 Seuillage Otsu

- Dans le script otsu.py, on cherche à minimiser la variance intra-classe.
  Si les niveaux de gris sont trop éparses, la méthode otsu ne semble pas très efficace à première vue.

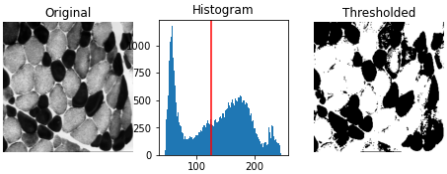

Figure 40: muscles

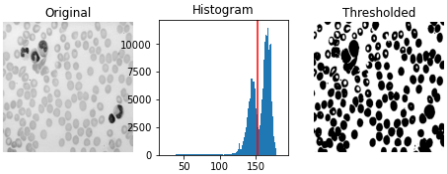

Figure 41: cellules

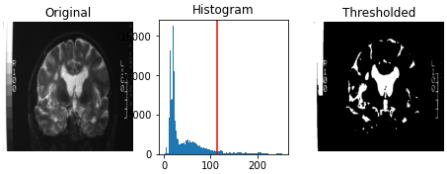

Figure 42: cerveau