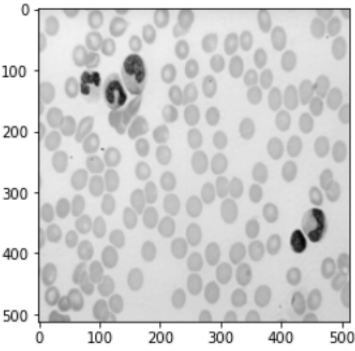
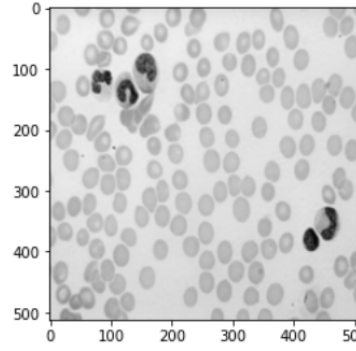
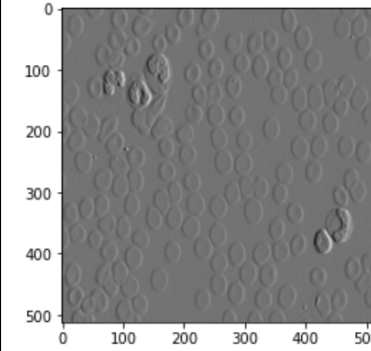
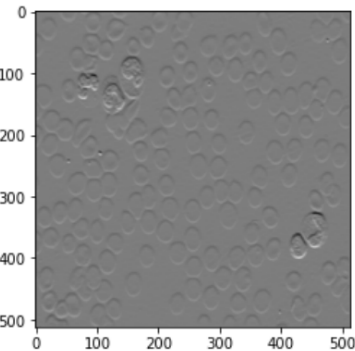
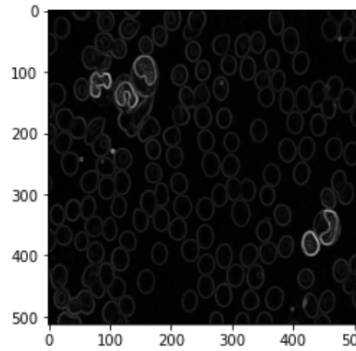
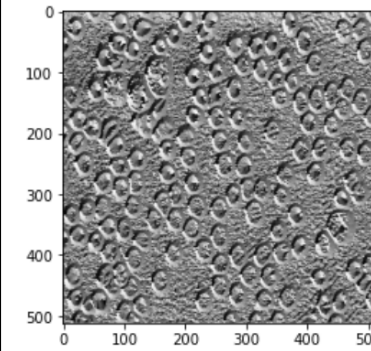
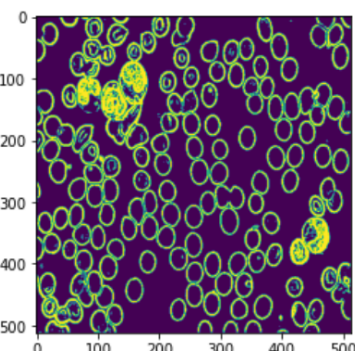
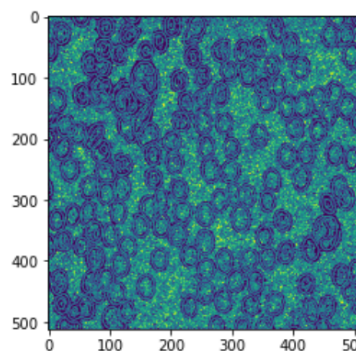
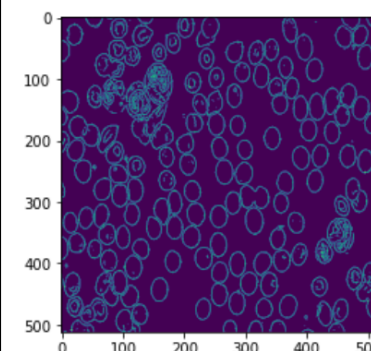


## TP 3 : Segmentation

### 1 Détection de contour

#### 1.1 Filtre de gradient local par masque

		
L'image cell.tif originale	L'image celle.tif après être passer par un filtre passe-bas	L'image cell.tif après avoir appliqué un filtre gradient vertical
		
L'image cell.tif après avoir appliqué un filtre gradient horizontal	La norme du gradient de cell.tif	La direction du gradient de cell.tif
		
La norme seuillée de cell.tif	Le maxima du gradient dans la direction du gradient	Le contour de l'image

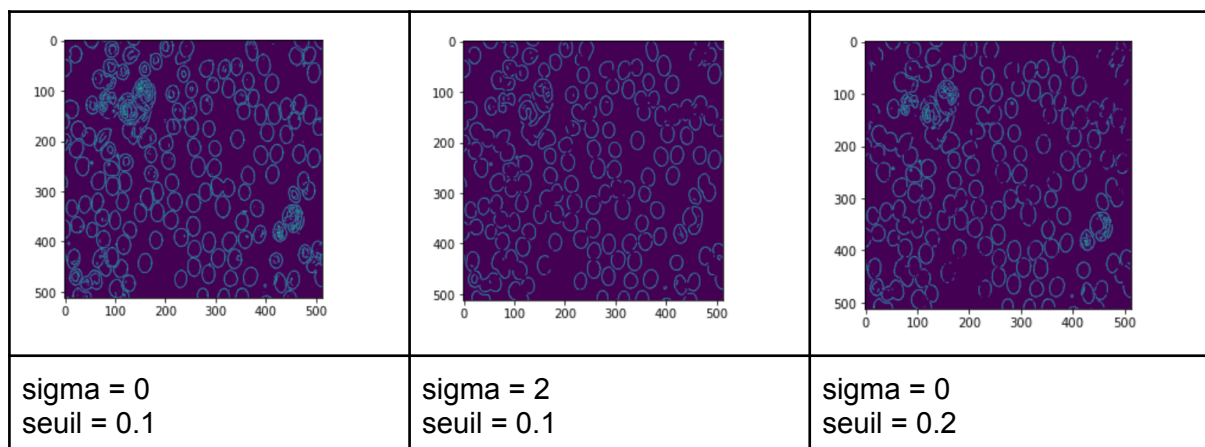
Le filtre de Sobel est couramment utilisé pour la détection de contour dans l'image. Par rapport à la simple différence entre 2 pixels voisins, il présente l'avantage d'être sensible à l'orientation. Il utilise 2 masques de convolution, un pour les contours verticaux et un pour les contours horizontaux.

On réalise souvent un filtre passe-bas à l'image pour réduire le bruit car les très grandes variations de norme du gradient perturbent l'algorithme.

En faisant varier le seuil, on observe que toutes les cellules ne sont pas forcément bien sélectionnées. En effet, certaines cellules apparaissent moins bien donc on a une norme plus petite qui peut passer sous le seuil de détection. Quand il y a beaucoup de bruit, on observe alors des objets qui ne devraient pas être détectés.

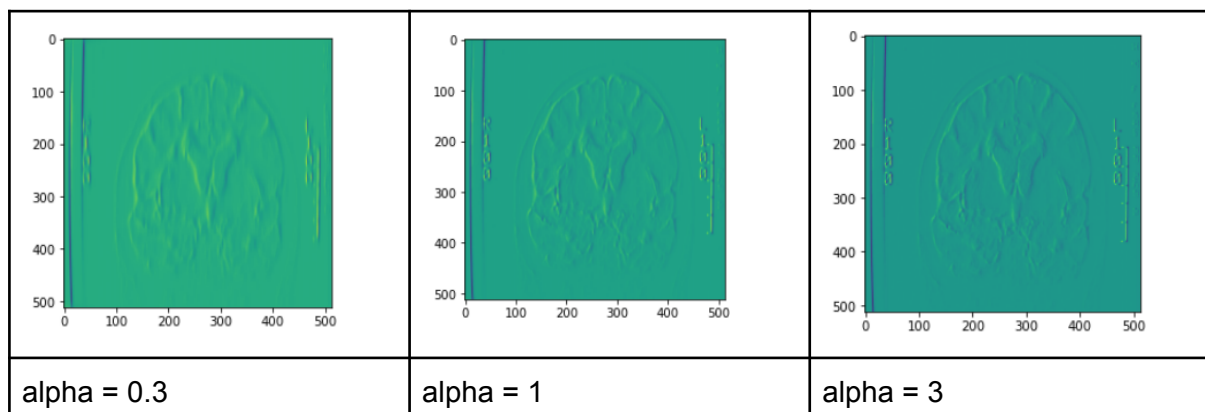
## 1.2 Maximum du gradient filtré dans la direction du gradient

Le critère de qualité optimisé ici est le contour des objets segmentés. En effet la méthode affine les contours sans risquer de perdre leur continuité. C'est une étape nécessaire si l'on veut par la suite vectoriser les contours.



On remarque que le flou du filtre passe bas et l'augmentation du seuil peut nuire à la continuité des contours. Un seuil élevé donne de la robustesse au bruit mais nuit à la continuité.

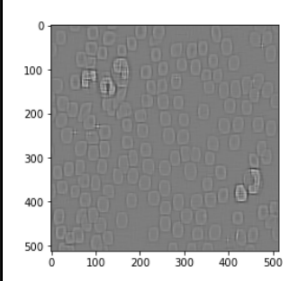
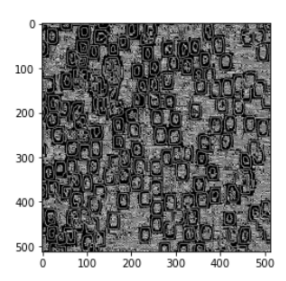
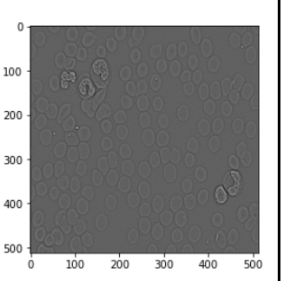
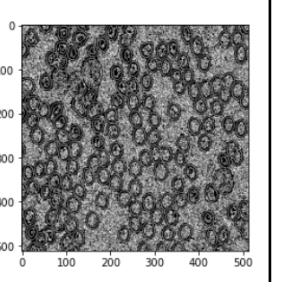
## 1.3 Filtre récursif de Deriche



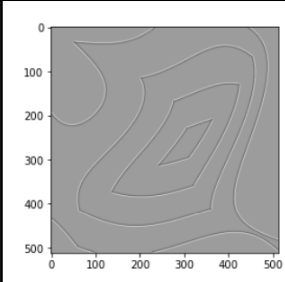


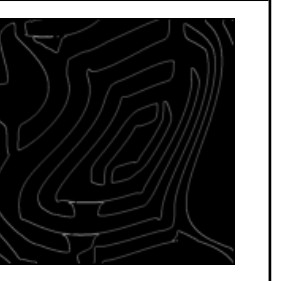
Augmenter alpha augmente la précision des contours mais augmente aussi la sensibilité au bruit. Le temps de calcul ne dépend pas de alpha, la complexité de l'algorithme ne dépend que de la taille de l'image.

Les filtres `dericheSmoothX` et `dericheSmoothY` servent de la même manière que pour `sobel`, à réduire le bruit et à améliorer la performance de l'algorithme en réduisant les variations importantes de niveau de gris.

#### 1.4 Passage par zéro du laplacien

			
Laplacien de cell.tif avec alpha = 0.5	Contour de cell.tif avec alpha = 0.5	Laplacien de cell.tif avec alpha = 3	Contour de cell.tif avec alpha = 3

On remarque qu'augmenter alpha permet d'améliorer grandement la précision des contours. Ceux-ci sont moins épais, donc plus fins et plus précis.

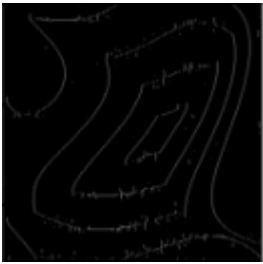
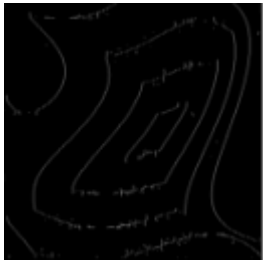
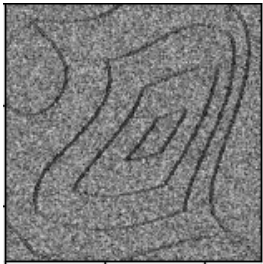
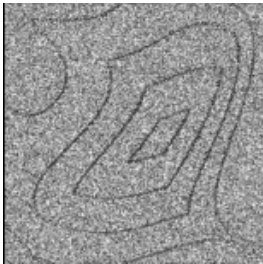
			
Laplacien de pyramide.tif avec alpha = 3	Contours de pyramide.tif avec alpha = 3	Laplacien de pyramide.tif avec alpha=0.5	Contours de pyramide.tif avec alpha=0.5

On remarque que la méthode du laplacien fait apparaître beaucoup de faux contours. Elle est moins satisfaisante que les méthodes précédentes pour l'image pyramide. Elle se révèle pour autant assez satisfaisante pour `cell.tif`.

#### 1.5 Changer d'image

Pour `pyra-gauss.tif`, qui est une image bruitée, il faut choisir un filtre peu sensible au bruit comme `Sobel` par exemple. On choisit après les paramètres adaptés.

On remarque aussi que la méthode du laplacien permet d'obtenir de très bons résultats.

			
Sobel sur pyra-gauss.tif avec seuil = 3	Sobel sur pyra-gauss.tif avec seuil = 4	Laplacien de pyra-gauss.tif avec alpha = 2	Laplacien de pyra-gauss.tif avec alpha = 3

## 2 : Seuillage avec hystérésis

### 2.1 : application à la détection de lignes

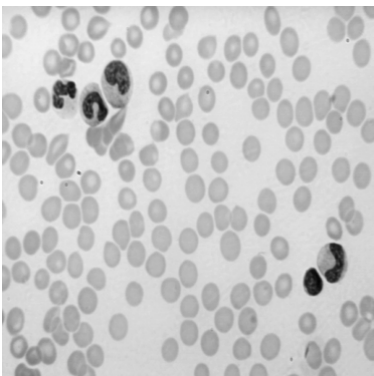
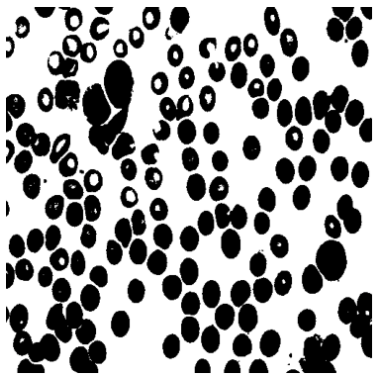
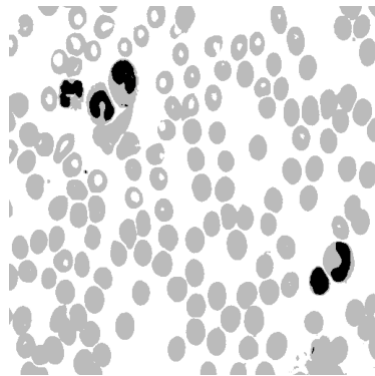
Le fait d'augmenter le rayon de l'élément structurant donne un résultat bien meilleur. Les lignes sont mieux détectées. Le temps d'exécution est cependant supérieur.

Avec un seuil trop élevé, beaucoup de contours disparaissent. En le baissant trop, on a au contraire trop de contours qui sont détectés. De même un seuil bas trop élevé fait disparaître les contours.

Les seuils entre 1 et 5 donnent des résultats satisfaisants.

## 3 Segmentation par classification : K-moyennes

### 3.1 Image à niveaux de gris

		
Image original	Image segmentées avec 2 classes	Image segmentée avec 3 classes

La segmentation avec 2 classes n'est pas suffisante pour détecter toutes les cellules de manière optimale. L'utilisation d'une troisième classe permet cependant de remédier à ce problème.

Pour muscle.tif, une segmentation en 3 classes est idéale. En effet, 2 classes ne sont pas suffisantes tandis qu'une quatrième classe scinderait les cellules elle-même. Le filtre médian permet d'enlever la texture des cellules et d'ainsi améliorer la segmentation.

### 3.2 Image en couleur

Appliquer une segmentation en 10 classes à l'image fleur.tif donne une image moins de nuance dans les couleurs. Par exemple, le centre des fleurs est souvent d'une unique couleur alors que l'image originale comporte de subtiles nuances. Les ombres sont moins bien représentées et des tâches de mauvaise couleur apparaissent.

On commence à obtenir une image similaire à l'original à partir de 20 classes.

Pour carte.tif, la solution est d'augmenter la valeur de  $n$  jusqu'à retrouver l'image originale.

### 4 Seuillage automatique OTSU

Le but de cet algorithme est de trouver de manière automatique un seuil qui permet de classer l'image en 2 canaux. Les pixels de valeur inférieure à ce seuil deviennent noir, les autres blanc.

Ici le critère qui est maximisé est la variance interclasse. Plus celle-ci est grande, plus les classes seront bien séparées.