TIP

Traitement d'image – Compte-Rendu

Adam Ben Ltaifa

Partie 1 - Basic Operations

1.Lire un histogramme

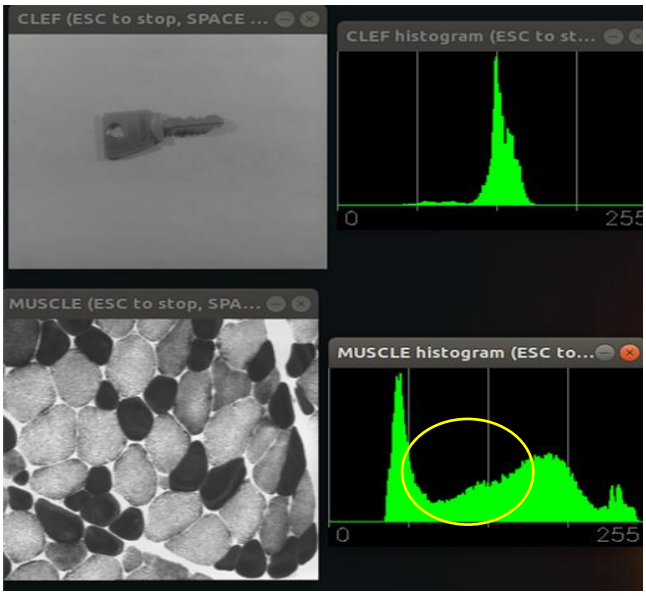


Figure 1. Histogrammes de deux images

- *Expliquer l'allure de chaque histogramme.*

Un histogramme comptabilise le nombre de pixels par niveau de gris. Un pixel proche de 0 est noir et un proche de 255 est blanc.

Première image, clef.bmp

Nous voyons bien que l'image est globalement **homogène**. Les pixels sont **concentrés** sur une plage de niveaux de gris restreinte, ce qui est représenté par un **pic** sur l'histogramme.

Le petit débordement dans les valeurs faibles (noir) correspond aux ombres de la clef.

Seconde image, muscle.bmp

Il est possible de découper l'histogramme en trois zones.

Le **premier pic**, le plus important, montre une concentration de pixels noirs correspondant aux **cellules sombres**.
La **section centrale**, plus diffuse, représente les **différents niveaux de gris des cellules blanches**, qui sont majoritaires.
Quant à la **dernière partie** de l'histogramme, celle-ci recense les pixels très blancs, soit les **espaces intercellulaires**.

- *Sur l'image muscle.bmp, quel est le type de cellule le plus « dense » (en termes d'occupation de l'image)*

Les cellules les plus **denses** sont les cellules musculaires blanches car l'**intégrale** de l'histogramme sur les niveaux de gris intermédiaires (cf. cercle jaune) est plus **élevée** que celle des cellules rouges. En d'autres termes, les **cellules blanches occupent la majorité** des pixels, d'où la large bande centrale.

- *Quel type de fibre est le plus homogène ?*

Les **fibres rouges** sont plus homogènes car sur l'histogramme les barres correspondant au noir sont **moins étalées**, centrées autour de la **même valeur**.

2.Opérations Arithmétiques

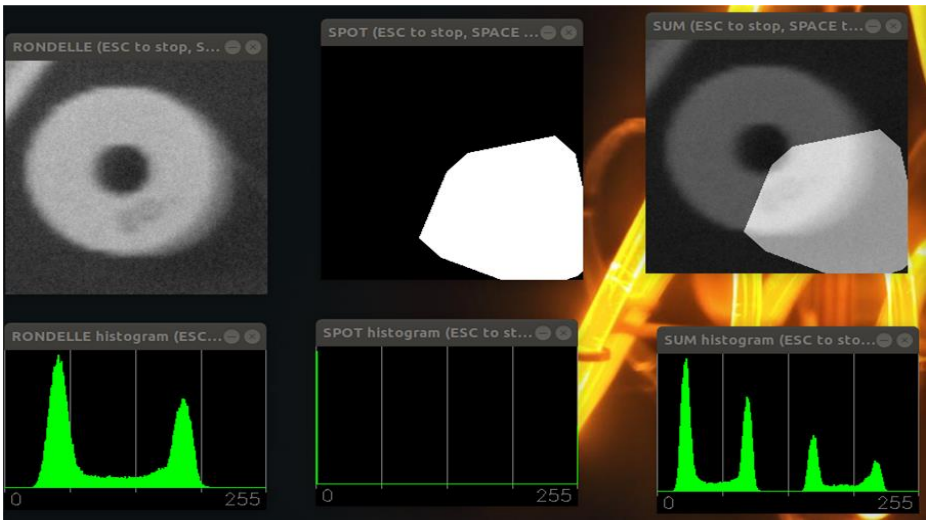


Figure 2.1 Rondelle, Spot blanc et leur somme après pondération

- *Pourquoi a-t-on multiplié par un scalaire avant de sommer les images ?*

Chaque pixel est codé sur 8 bits d'intensité de gris.
Il peut donc prendre des valeurs allant de **0 à 255 au maximum**. On cherche donc à éviter la **saturation**.

À l'aide du bloc *Multiply by Scalar*, on va **réduire la valeur des pixels** pour que la somme (image résultante) **ne soit pas saturée** et **n'excède pas 255**.

Histogramme de l'image résultante (le troisième)

Nous y distinguons 4 teintes majoritaires de gris différentes.

Les **deux premiers pics** correspondent à la zone **non affectée par le spot blanc**, mais leur valeur de gris a été **diminuée de moitié** environ par le bloc *Multiply by Scalar*. Nous retrouvons donc la **même allure** que l'histogramme original (le premier), mais **décalé vers la gauche et donc vers le noir** (leur nombre a également diminué car un quart de la rondelle a été touchée par le cube).

Pour ce qui est des deux autres pics, ceux-ci correspondent aux autres zones affectées par le spot blanc (rondelle et fond noir).

- *On retire le bloc Multiply by Scalar. Expliquer la forme de l'histogramme.*

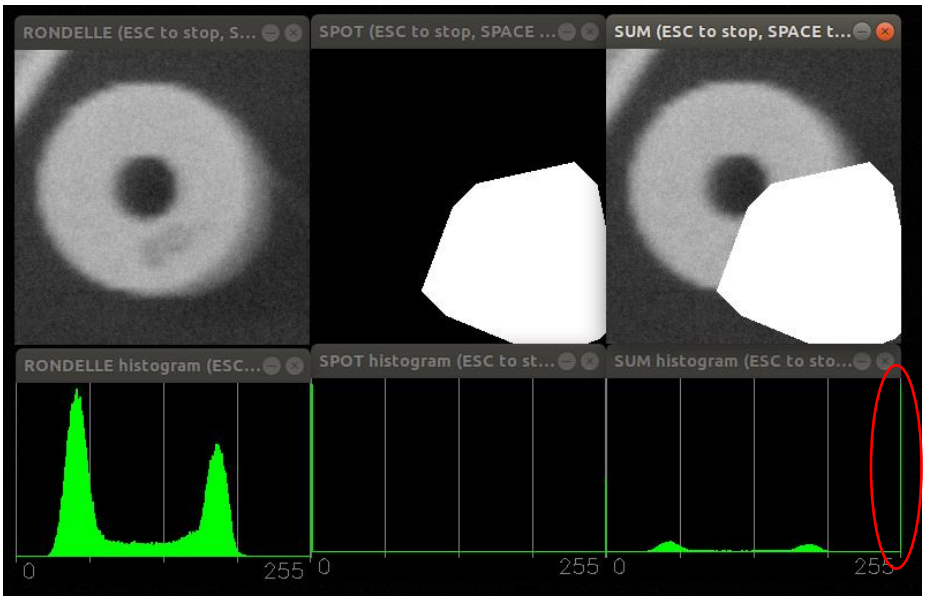


Figure 2.2 Somme Rondelle et Spot blanc avec saturation

On ne distingue plus que trois nuances : **gris foncé**, **gris clair** et un **blanc saturé** avec un **pic à 255**.

Lors de l'addition **tous les pixels sommés au spot excèdent 255**, ce qui s'observe par le **pic (cercle rouge)**. Les deux autres lobes sont à la même position car additionnés avec 0.

La forme **aplatie** de l'histogramme est en réalité un **problème d'échelle**. Nous avons simplement un très grand pic de blanc à 255, bien supérieur à celui des deux autres lobes.

- *Est-il possible de prédire la forme de l'histogramme résultant en analysant les deux autres ?*

Il est impossible de prédire à quoi peut ressembler l'histogramme car celui-ci ne prend pas en compte l'information spatiale, c'est-à-dire l'endroit où se trouvent les différents niveaux de gris dans l'image.

3.Négatif d’une image

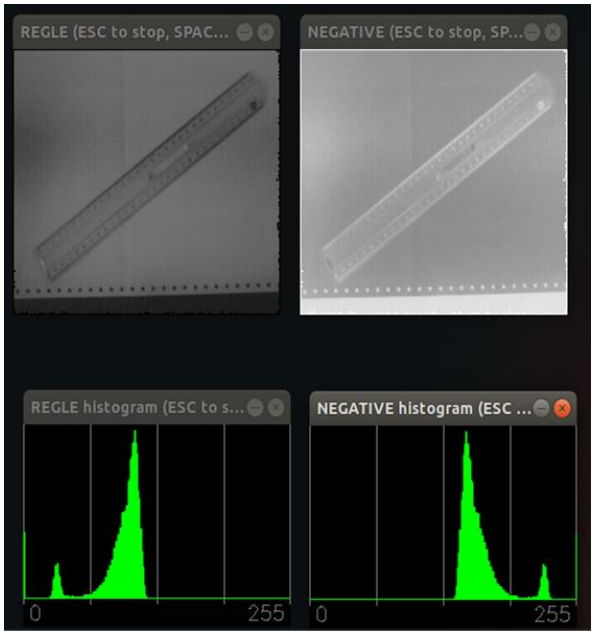


Figure 3. Négatif d’une règle

- *Expliquer l’histogramme.*

Le négatif d’une image s’obtient en **soustrayant** celle-ci à une image entièrement blanche, dont tous les pixels ont une valeur de **255**. Par cette opération mathématique, nous obtenons **le symétrique vertical** à celui de l’image originale.

4.Quantification

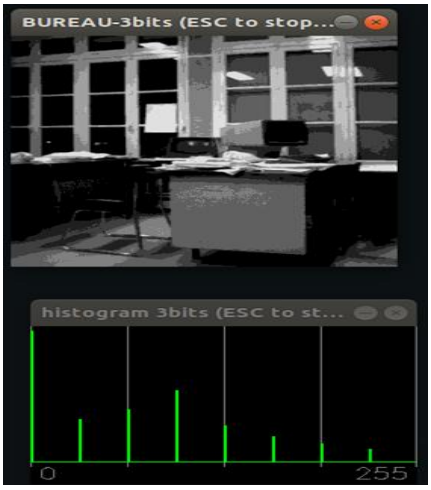


Figure 4.1 Image d’un bureau quantifiée sur 3 bits

- *À la vue de cet histgramme, quelles sont vos conclusions ?*

Suite à la quantification, nous perdons de la qualité d'image car nous **perdons des plages de niveaux de gris**. Les **constrates** sont donc plus marqués (saut entre plages de gris plus brusques).

1 bit → 2 niveaux de gris

3 bits → 8 niveaux de gris

5 bits → 32 niveaux de gris ce qui donne une image satisfaisante à l'œil car l'homme ne distingue que 30 niveaux de gris.

Nous observons 8 pics car sur 3 bits nous avons bien 8 niveaux de gris.

- *Analyser les 'most and less significant binary plane'*

Nous appliquons un **masque** b1000.0000 pour ne garder sur chaque pixel que le **bit 7**, qui est le **plus signifiant**. Nous avons un bit actif, qui nous donne deux niveaux de gris possibles : 128 et 0.

Ce bit actif est le '*most significant binary plane'*.

Pour obtenir le *'less significant binary plane'*, nous utilisons la même logique mais en appliquant un masque b0000.0001. Ainsi, il n'y a que deux niveaux de gris possible : 0 et 1.

N.B. Comme les gris de valeurs 0 et 1 sont très proches, nous procédons à un *scaling* afin de voir la différence.

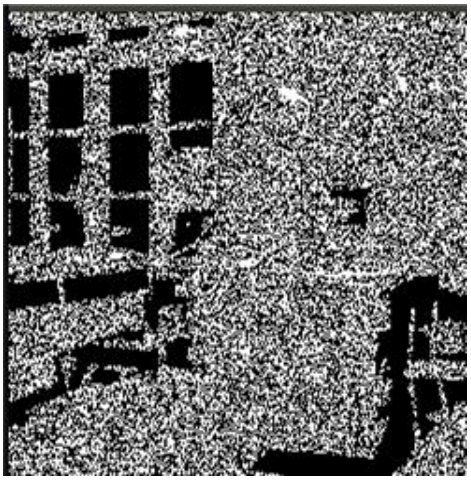


Figure 4.2 Plan binaire le moins signifiant après *scaling*

Partie 2 – Transformation d'histogramme

1.Amélioration d'image par modification d'histogramme

- *Donner la formule permettant ce rehaussement.*

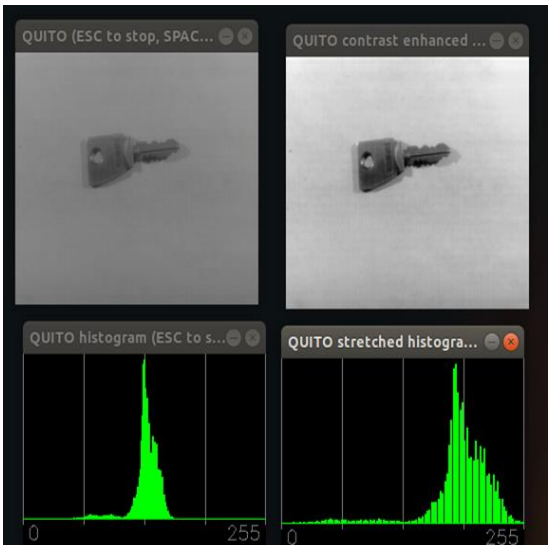


Figure 5.1 Recadrage linéaire

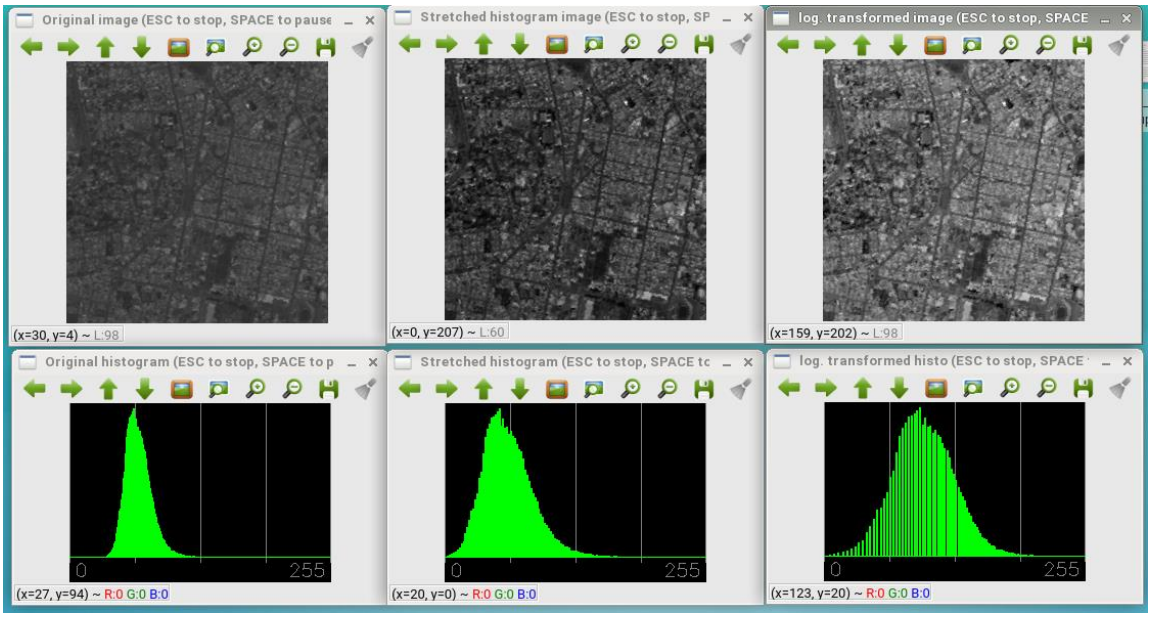


Figure 5.1 Recadrage linéaire et transformation logarithmique

On note X l'image originale et X' l'image modifiée. X_{max} la valeur maximum de l'image et X_{min} sa valeur minimale. La formule du recadrage linéaire est donnée par :

$$X' = (255/(X_{max} - X_{min}))*(X - X_{min})$$

Sur la première image (clef.bmp), nous réalisons une opération de **recadrage linéaire**, ou *stretching*.

Sur la seconde (quito.bmp), nous appliquons un stretching suivi d'une transformation logarithmique.

Le « **stretching** » permet d'étirer l'histogramme artificiellement de manière à ce qu'il occupe l'entièreté de l'intervalle de valeurs. Cependant, ce traitement laisse des zones inoccupées sur l'histogramme, ce qui implique que des niveaux de gris sont inutilisés dans l'image finale.

Cela permet **d'améliorer les contrastes** d'une image présentant beaucoup de petits détails.

La **transformation logarithmique** permet de simuler un **histogramme à distribution normale** et de mettre en évidence la répartition des valeurs, notamment des plus faibles.

2.Égalisation d'histogramme

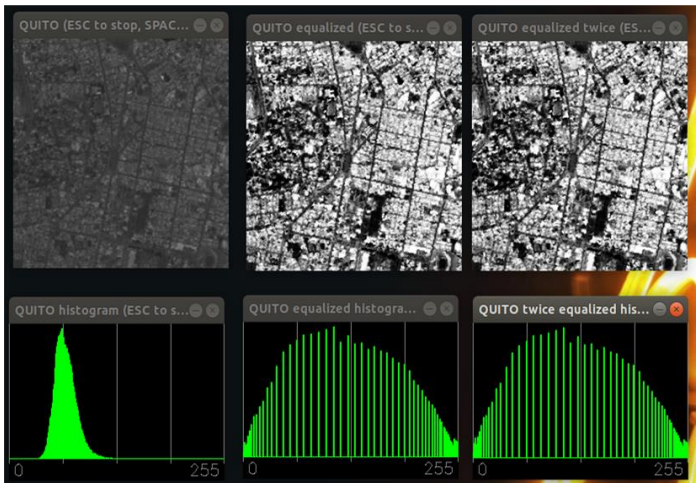


Figure 6 Égalisation d'histogramme

- *Commentez les histogrammes.*

Suite à l'opération d'égalisation, nous passons d'une image « voilée » de gris à une image **davantage contrastée**, avec du noir foncé et du blanc marqué.

L'égalisation permet de « **répartir les pixels** » sur chaque niveau de gris. Chaque pixel de l'image va subir une transformation qui modifier son niveau de gris, de manière à observer une distribution uniforme de l'histogramme.
Ainsi, les niveaux de gris à **forte population sont étalés** tandis que ceux à plus **faible nombre** de pixels sont **regroupés**.

Cela permet de **mettre en évidence les détails qui étaient masqués** dans des régions presque **homogènes** de l'image.

- *Quel est l'effet de deux égalisations successives ?*

La seconde égalisation n'a aucun impact sur l'image. En effet, vu que l'histogramme cumulé (qui permet de faire l'égalisation) utilisé est le même, il est logique d'obtenir le même résultat.

Ainsi, l'égalisation est une **transformation idempotente**.

3.Seuillage d'histogramme

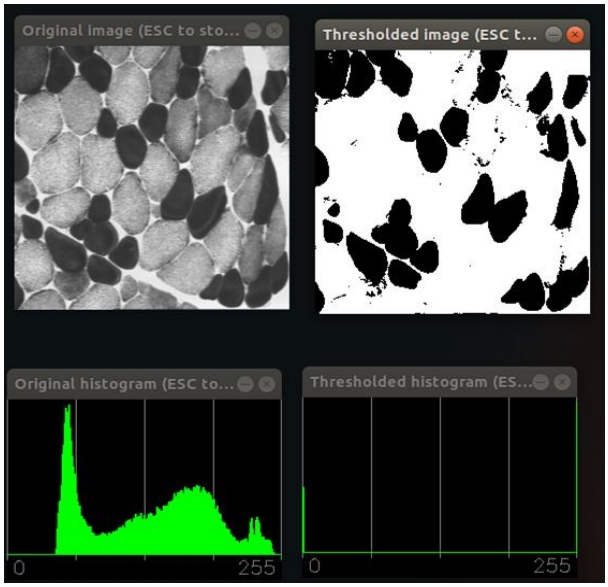


Figure 7 Seuillage

- *Commentez l'histogramme*

Le ***thresholding*** **ou seuillage** divise l'histogramme en deux parties séparées. La valeur de séparation est le paramètre appelé **threshold ou seuil**. Tout ce qui se trouve à **gauche du threshold** est ramené à la valeur extrême **0** (noir), et tout ce qui se trouve à **droite** à la valeur extrême **255** (blanc).

Dans notre exemple, nous avons utilisé un threshold à environ 90 de façon à isoler les cellules rouges, représentées en noir sur l'image. Ainsi, tous les pixels inférieurs à cette valeur sont devenus noirs et ceux au-dessus devenus blancs, ce qui permet de libérer l'image des cellules blanches.

Nous pouvons comparer cela à une **forme de quantification** : les valeurs ''continues'' (bien que dans notre cas elles soient discrètes) tombées entre deux seuils sont ramenées à ceux-ci.

Cela explique la forme de l'histogramme résultant.

Partie 3 – Filtres linéaires et non linéaires

1.Filtre moyenneur (Moyenne glissante)

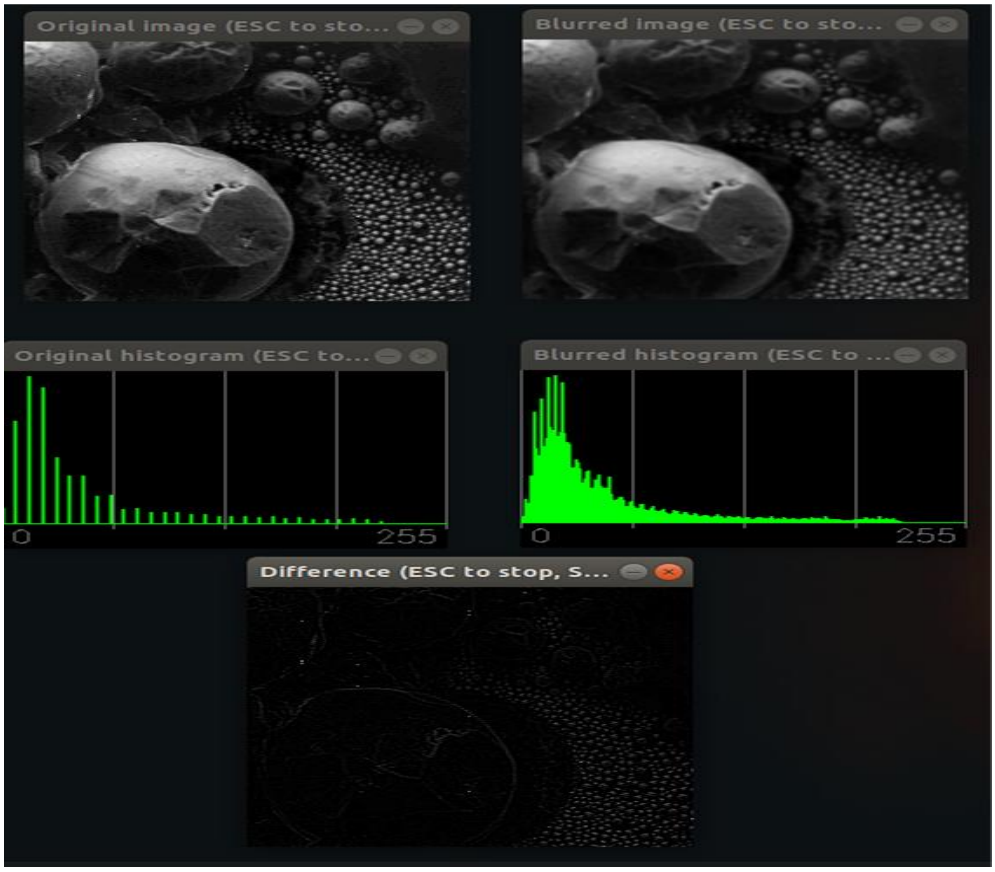


Figure 8.1 Filtre moyenneur 3x3 et Différence

- *Pouvez-vous déduire le nombre de bits utilisés formant l'image originale ?*

Il y a 29 pics distincts sur l'image originale, soit 29 niveaux de gris.
Ainsi, un codage sur 5 bits est suffisant (2^5 = 32)

- *Expliquer la forme de l'histogramme après application d'un filtre moyenneur 3x3*

Premièrement, on reconnaît la **même enveloppe globale**.
Cependant, l'opération de moyennage a **créé de nouvelles valeurs** qui n'étaient pas dans l'image originale. Cela engendre de **nouveaux niveaux de gris**, et explique pourquoi l'histogramme est plus « dense ».

- *Expliquer l'allure de l'image résultant de la différence entre l'originale et la moyennée.*

L'image différence représente l'**erreur d'approximation**, autrement dit les détails perdus après l'opération de moyennage. Chaque pixel représente l'intensité du niveau de gris perdu lors du filtrage, notamment ceux des contours.

- *Comparer l'image à laquelle on aurait appliqué 2 fois un filtre moyenneur 3x3 et la même avec un filtre 5x5*

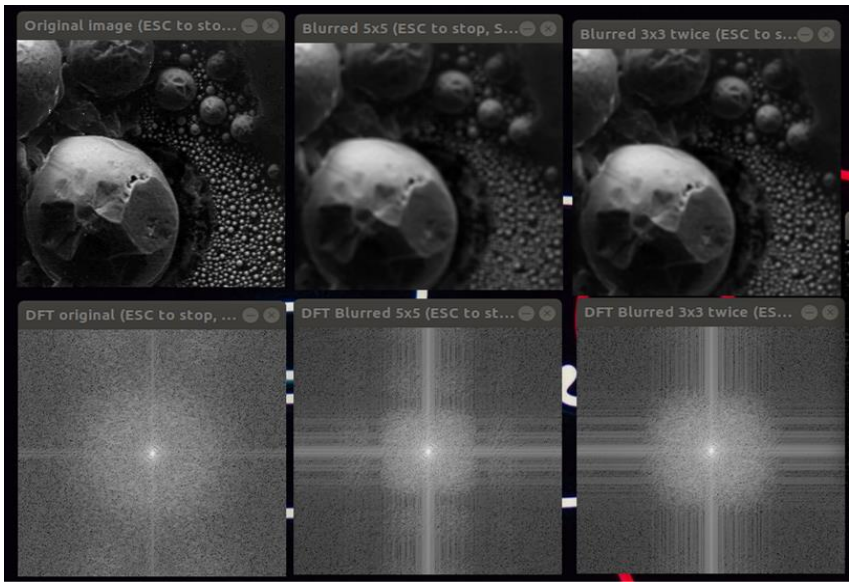


Figure 8.2 Double filtrage 3x3, Filtrage 5x5 et Transformée de Fourier Discrète

Les images résultantes ne sont pas identiques.

À l'œil, l'image « 3x3 twice » est plus **nette** que celle en « 5x5 ».
En effet, ce dernier moyennage est plus brutal que l'autre (ex : en faisant la moyenne de ma note et de celle de 2 autres élèves, il y a plus de chances que la moyenne soit représentative de mon niveau réel que si je le faisais avec toute la classe).

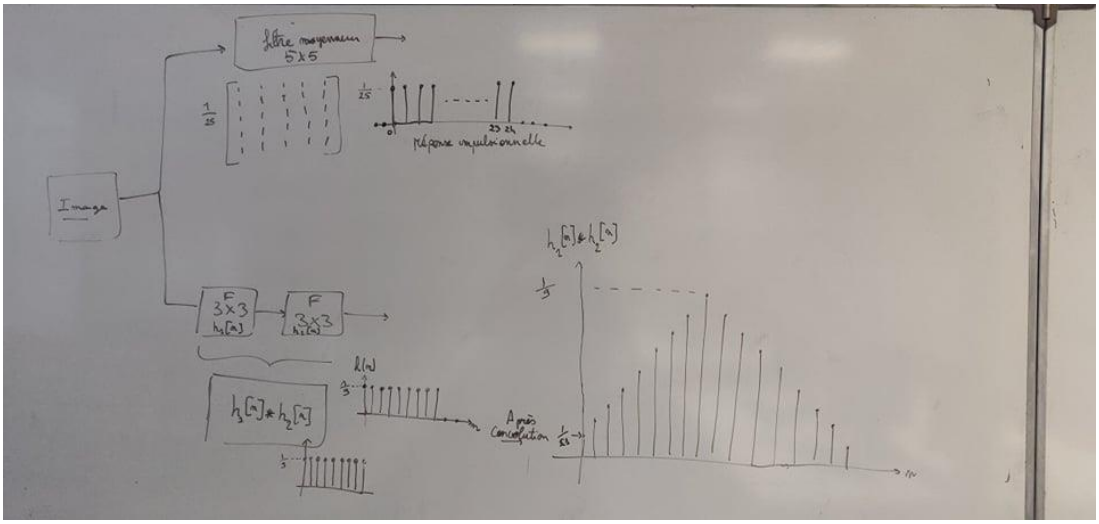


Figure 8.3 Comparaison de l'allure des filtres 2 fois 3x3 et 5x5

Le **filtrage 5x5 retire** plus efficacement les **hautes fréquences** (contours), ce qui se voit sur l'image mais aussi sur la DFT (Discrete Fourier Transform).
En effet, on remarque que les fréquences les plus éloignées du centre (et donc les plus élevées) sont moins nombreuses (moins de lignes verticales et horizontales) que pour l'image filtrée deux fois en 3x3.

2.Filtrages non-linéaires

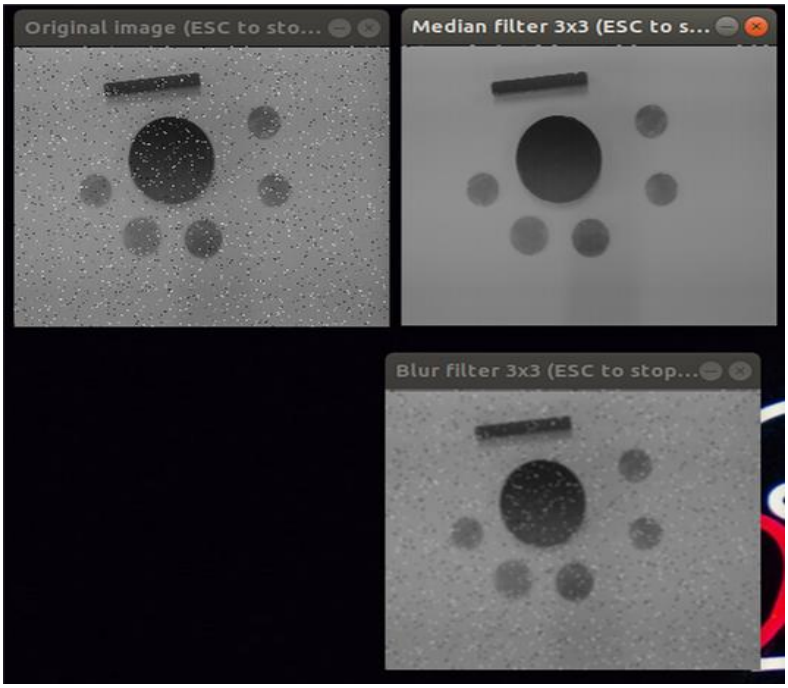


Figure 9.1 Comparaison filtre médian et filtre moyenneur sur bruit poivre et sel

- *Comparer les effets du filtre moyenneur et du filtre médian sur une image affecté par du bruit « poivre et sel »*

Nous avons pour image originale une image dégradée par 5% de bruit **''poivre et sel''**, c'est-à-dire que des pixels ont été placés à des valeurs très faibles ou très hautes.

Nous voyons que le **filtre moyenneur** (en bas à droite) donne un résultat peu satisfaisant. En effet, au lieu de faire disparaître ces valeurs anormales, il les **intègre dans son calcul**. De fait, ces valeurs extrêmes vont affecter la moyenne de leurs voisins, et **dégrader ainsi l'ensemble de l'mage**, le tout sans que le bruit ait été retiré.

Le **filtre médian** (en haut à droite) est une bien meilleure solution, puisque pour chaque pixel donné et son voisinage direct, il va remplacer ce pixel par la **valeur médiane**. Cela permet de **retirer les valeurs aberrantes** trop faibles ou trop hautes et ainsi d'obtenir une image exploitable.

- *Même chose sur une image avec du bruit uniforme.*

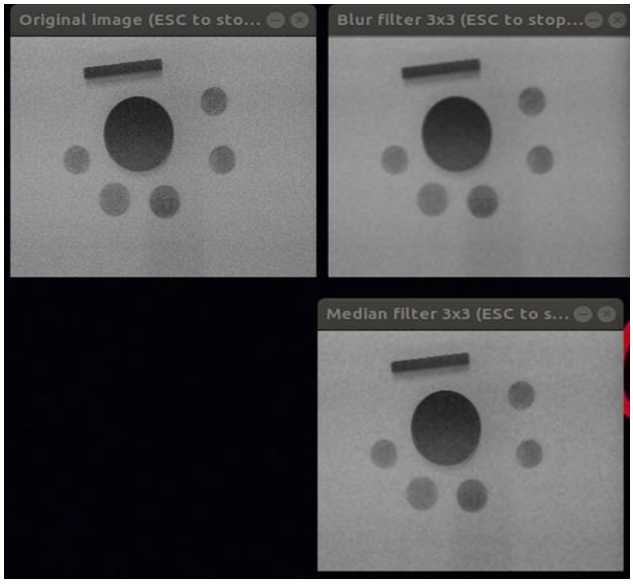


Figure 9.2 Comparaison filtre médian et filtre moyenneur sur bruit uniforme

Dans ce cas-ci, le filtre moyenneur semble légèrement plus avantageux que le filtre médian car il diffuse à toute l'image un bruit déjà uniformément réparti et dont les valeurs sont proches de l'image originale. Toutefois, cela dégrade également la netteté.
Le filtre médian, lui, peine davantage car le bruit n'est pas aberrant ; le filtre **sélectionne donc parfois la valeur de bruit** au lieu de le retirer.

- *Répéter les étapes avec les images de fissures.*

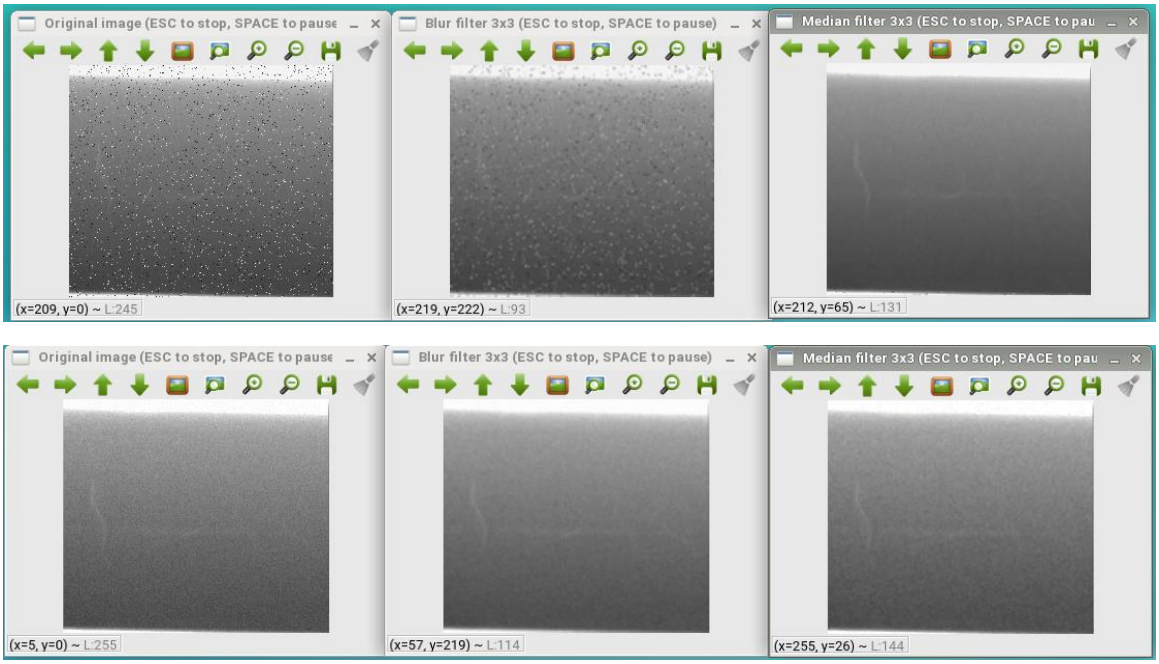


Figure 9.3 Mêmes comparaisons avec une image de fissure

Nous pouvons donc conclure que dans le cas d'un bruit **poivre et sel**, on utilisera un **filtre médian**. Dans le cas d'un **bruit uniforme**, plutôt le **filtre moyenneur**.

3.Filtres linéaires détecteurs de contours

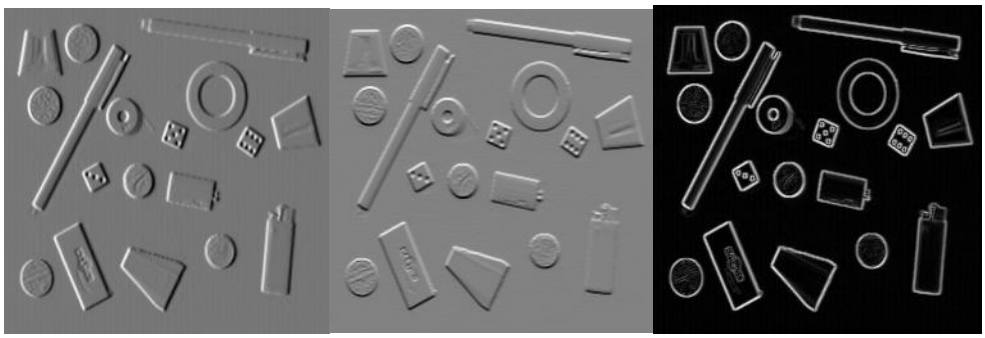


Figure 10 Gradient en X (Sobel), Gradient en Y (Sobel), Somme des gradients

- *Commenter les résultats d'une utilisation du filtre de Sobel sur l'image*

Nous voyons sur la première image que le **gradient en X** montre les **détails verticaux**, car en parcourant horizontalement l'image, il détecte les changements, qui sont donc verticaux.
Le **gradient en Y**, lui, met en évidence les **détails horizontaux**.

Les **contours blancs** sont issus de **dérivées positives** (on « rentre » dans l'objet) tandis que les **contours noirs** viennent de **dérivées négatives** (on en « sort »).
Le **gris** représente la dérivée **nulle**.

Ces opérations ont été réalisées à l'aide de filtres de Sobel, dans lesquels nous pouvions choisir sur quel axe calculer le gradient.

La **somme des deux gradients**, passés au préalable en valeurs absolues, permet d'obtenir l'ensemble des contours des objets, et peut donc **servir de détecteur de bords**.

4.Opérateur de Canny

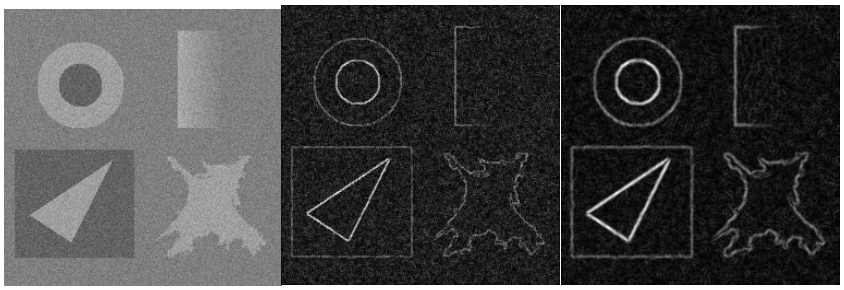


Figure 11.1 Détection de bords avec Sobel et Sobel+Filtre Gaussien

- *Quelle image a des bords plus visibles ?*

L'image issue du traitement **Gaussian Filter + Sobel** est plus visible, que celle uniquement avec Sobel.

En effet, il est intéressant de **filtrer le bruit gaussien** avant d'évaluer les contours avec Sobel, et ainsi **éviter de détecter les faux contours issus du bruit**.

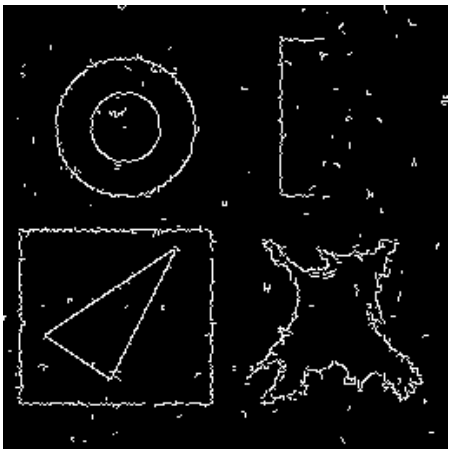


Figure 11.2 Filtre de Canny

- *Décrire les étapes du filtre de Canny.*

Le **filtre de Canny** est plus évolué et précis pour détecter les bords, notamment sur les images complexes, grâce à l'ajout de quelques étapes :
(cf. https://en.wikipedia.org/wiki/Canny_edge_detector)

Filtrage Gaussien

→ Élimine le bruit gaussien

Filtrage par le filtre de Sobel (ou Prewitt, Roberts)

→ Détecte les gradients de l'image sur chaque axe, et ainsi la direction des bords

Suppression des non-maxima

→ Permet de déterminer si un bord l'est véritablement en vérifiant si un pixel est son maximum local suivant une direction donnée

Double seuillage

→ Les pixels ayant été retenus jusqu'ici sont divisés en trois catégories grâce à un double seuillage. On pose **(T1, T2) les thresholds**. Nous avons d'un côté les pixels « **à garder** » à très haut gradient (supérieurs à T1), les pixels « **indécis** » à gradient moyen (entre T1 et T2) et les pixels « **à supprimer** », à gradient faible (inférieurs à T2).

Edge Tracking Hysteresis

→ Pour les pixels "indécis", soit entre T1 et T2, il nous faut vérifier leur entourage : s'ils contiennent **dans leur entourage un pixel fort,** alors ils font eux aussi partie d'un bord et son donc représentés en blancs sur l'image. Sinon, ils sont supprimés.

5.Opérateur de Laplace

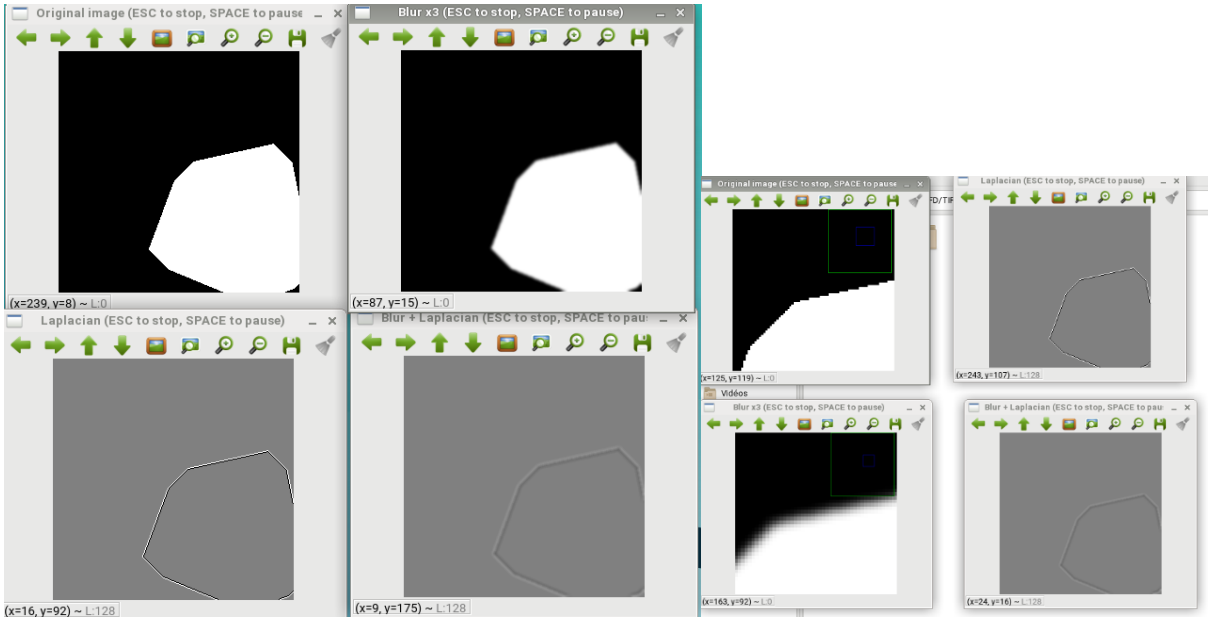


Figure 12.1 Comparaison d'un laplacien sur une image nette et une image floutée

Le **laplacien** (cf. image en bas à gauche) appliqué à l'image orignale **détecte bien les contours** (bords blancs si dérivée seconde très forte et bords noirs si dérivée très faible) car la **transition est brute** au niveau des bordds.
Ainsi, cet opérateur est très efficace pour détecter les contours dans ce genre de cas.

Suite à l'application triple d'un **filtre moyenneur**, les **contours sont maintenant flous**. Cela explique donc pourquoi le laplacien peine davantage à les mettre en valeur (gris clair et gris foncé) car les **dérivées secondes sont plus proches** de zéro dans ce cas.

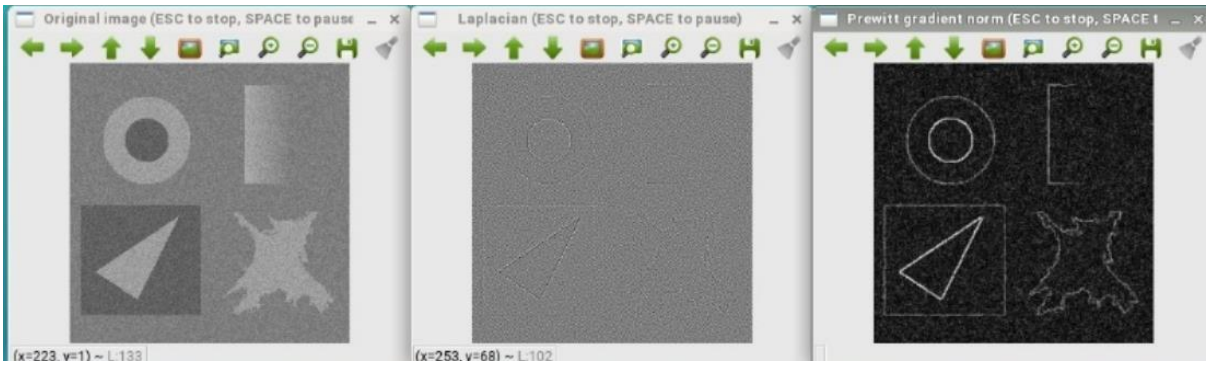


Figure 12.2 Comparaison laplacien et filtre de Sobel

- *Sur quelle image filtrée détectez-vous le mieux les bords et pourquoi ?*

Les bords sont bien plus visibles avec Sobel qu'avec le laplacien car le laplacien est **plus sensible au bruit** puisqu'il utilise des **dérivées secondes**.

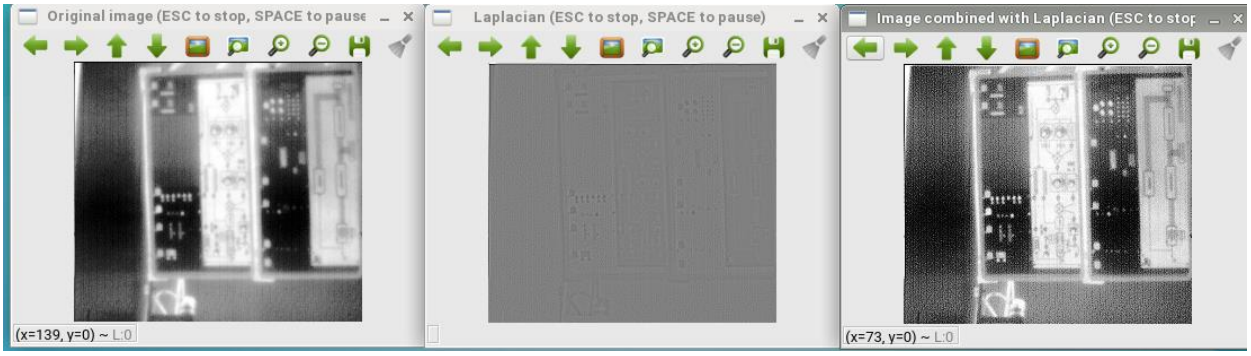


Figure 12.3 Addition d'un laplacien à 60% de l'image originale

- *Additionnez le laplacien d'une image à 60% de l'image originale. Commentez.*

L'addition d'une image et de son laplacien permet de réaliser une opération **d'amélioration de contours**.

(N.B. Dans ce cas-ci, nous avons d'abord calculé le laplacien, puis effectué son négatif, que nous avons ajouté à l'image originale. L'utilisation du négatif s'explique par le fait que la librairie utilisée pour calculer le laplacien utilise un masque opposé à celui du cours)

Partie 4 – Morphologie Mathématique

Figure 13.1 Dilatation et Érosion

Soit un élément structurant se déplaçant sur toute l'image.

Dilatation : Si l'intersection entre notre élément structurant et l'image contient au moins un pixel blanc (pixel objet), alors le pixel central devient blanc.
→ Permet de boucher les trous plus petits que l'élément structurant.

Érosion : Opération duale de la dilatation Si l'intersection n'est pas totale entre l'élément structurant et l'objet, alors le pixel central est remplacé par du noir.
→ Élargit les canaux et les trous de l'image en éliminant les composantes connexes plus petites que l'élément structurant.

Figure 13.2 Érosion du négatif d'une image

- *Faire le négatif de l'image originale, puis l'éroder. Conclure.*

Nous remarquons que les deux images résultantes sont identiques en valeur absolue, mais l'une étant le négatif de l'autre.

Cela s'explique par le fait que qu'éroder le négatif revient à former le négatif de la dilatation :

$$\text{Dilatation}(x) = \text{Négatif}(\text{ érosion } (\text{ négatif } (x)))$$

- *Dilater la sortie de l'érosion, et éroder la sortie de la dilatation. Comment appelle-t-on ces combinaisons ?*

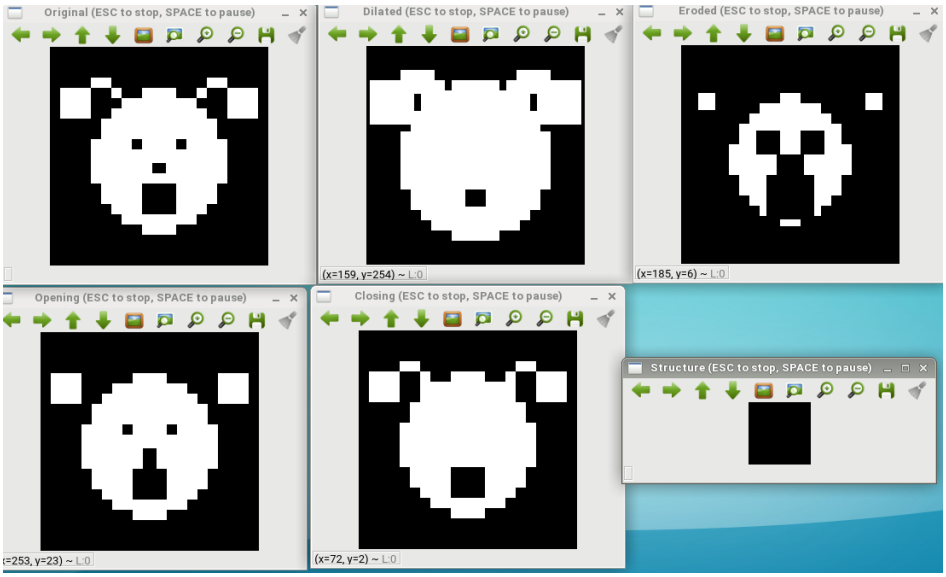


Figure 13.3 Ouverture et Fermeture

La composition d'une dilation morphologique avec l'érosion par le même élément structurant **ne produit pas, en général, l'identité**. Idem dans l'autre sens.

Ces transformations ne sont **pas bijectives**, les opérations de dilatation/érosion ne sont pas réciproques. Une image dilatée ou érodée peut avoir **plusieurs images antécédentes différentes**.

D'après Wikipédia, les noms de ces compositions sont :

Ouverture
$\mathrm{Ouverture(x)} = \mathrm{Dilatation(Érosion(x))}$
→ Permet de retirer le bruit extérieur à l'objet

Fermeture
$\mathrm{Fermeture(x)} = \mathrm{Érosion(Dilatation(x))}$
→ Permet de fermer les petits trous de l'objet (typiquement des petits points noirs dans un objet blanc)

Partie 5 – Segmentation d'images

1.Seuillage automatique

- *Commentez le fonctionnement du seuillage automatique.*

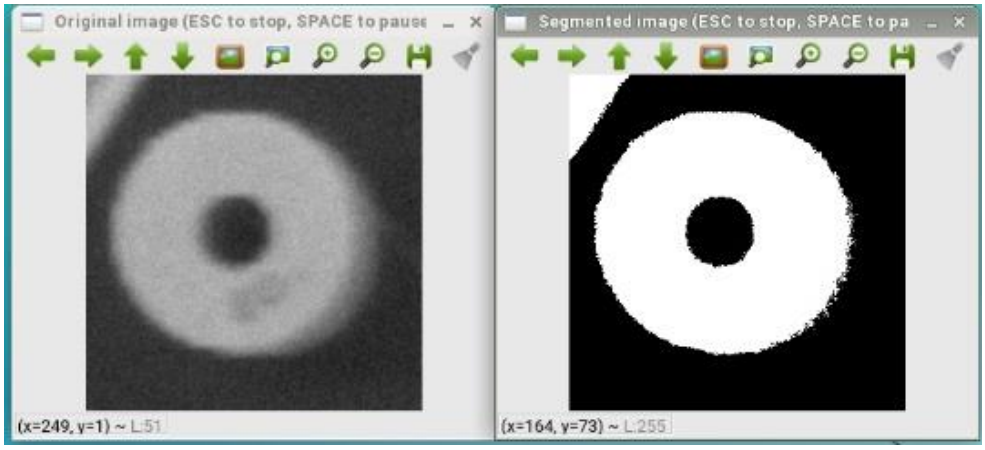


Figure 14.1 Seuillage automatique

Le **seuillage automatique** (***automatic thresholding***) repose sur l'algorithme d'**Otsu**, lui-même adapté de l'algorithme de Fisher qui permet par une méthode itérative de déterminer le meilleur seuil permettant de partitionner l'histogramme en deux sous-classes, de façon à ce que leur **variance intra-classe soit minimale** (cf. cours TIP 117).

Une fois ceci fait, on associe les valeurs de la première classe à la valeur 0 et celles de la seconde à 255.

2.Segmentation *Split and Merge*

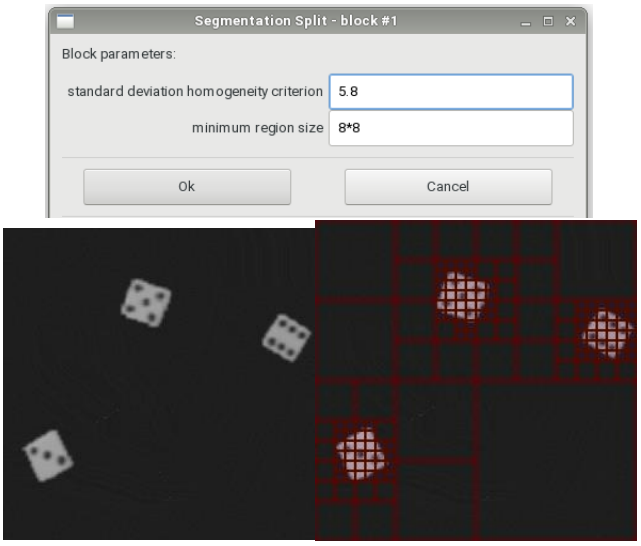


Figure 14.2 Split

- *Expliquer le résultat de la segmentation*

Cette méthode de segmentation permet d'identifier les zones homogènes de l'image. Elle suit l'algorithme suivant :

On initialise le carré à l'ensemble de l'image.

1/ Le contenu du carré est-il **homogène** selon notre critère ? C'est-à-dire, est-ce que **l'écart-type des pixels de l'image est inférieur au seuil** que nous avons fixé ?

2/ **Oui :** Cette portion de l'image est considérée comme homogène, l'algorithme termine pour cette portion.
Non : Découpons ce carré en **quatre parties** égales, et réalisons la même opération sur **chacune** d'entre elles.

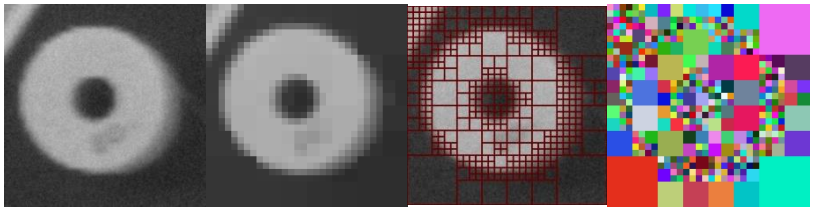


Figure 14.3 Split & Merge

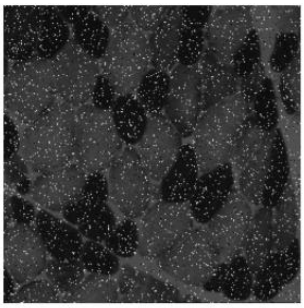
Le « ***merge*** », ou **fusion**, est l'opération **d'agrégation des pixels « similaires »**, c'est-à-dire qui respectent le **second critère d'homogénéité** (qui peut être différent du split).

Sur Starling, ce bloc prend deux entrées : l'image originale, et l'ensemble des régions après le split, qui est en réalité un arbre. En parcourant cet arbre, les pixels similaires sont rassemblés et considérés comme faisant **partie d'un seul objet**.
Ainsi, cela permet de **séparer les éléments distinctifs** d'une image, comme lez zones avec la même texture, forme etc.

(N.B. Dans cet exemple, nous avons choisi un écart-type trop élevé, ce qui fait que les portions de l'image sont trop distinctes.)

EXERCICES FINAUX

Exercice 1



Original image

Result to be obtained

Objectif : mettre en évidence les cellules rouges et les cercler de blanc tout en supprimant les cellules blanches. Voici ce que nous proposons.

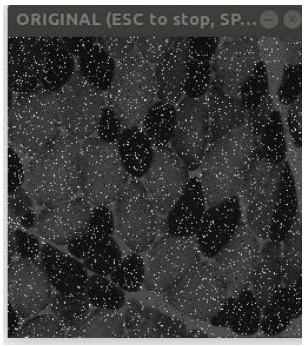
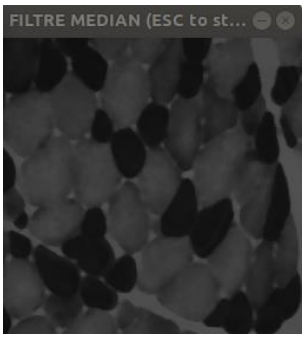
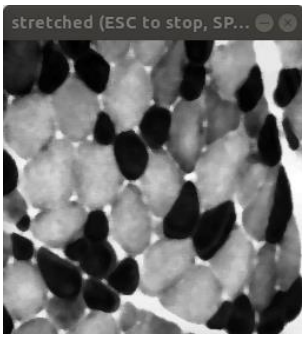


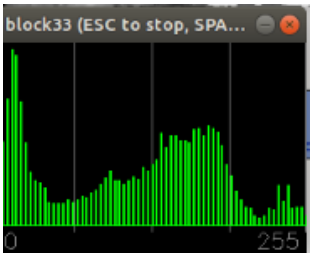
Image originale, dégardée par du bruit poivre et sel



Filtre Médian : Retire le bruit poivre et sel



Réhaussement d'histogramme : Améliore le contraste de l'image afin de mieux différencier les cellules



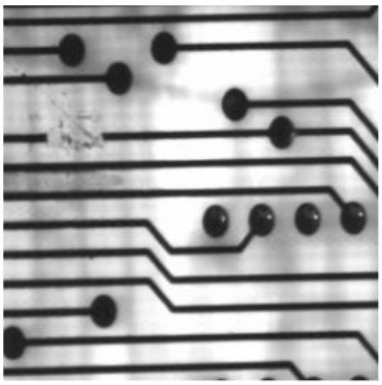
Histogramme réhaussé

Seuillage manuel : Nous choisissons une valeur qui donne l'allure que nous souhaitons obtenir. Nous ne gardons que les cellules noires et éliminons les blanches.

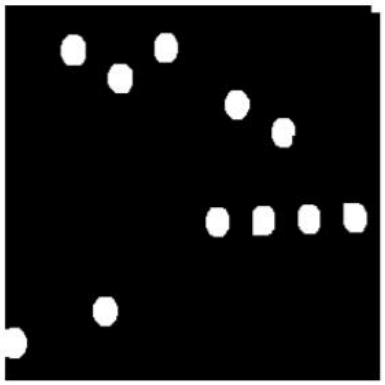
Laplacien : Permet d'obtenir les contours des éléments de l'image

Dilatation : Épaissit les contours et soude les formes proches

Exercice 2



Original image



Result to be obtained

L'image originale représente un circuit imprimé en noir sur fond grisâtre non homogène, avec des traits verticaux et diagonaux.
Objectif : Obtenir les points nodaux, en blanc sur fond noir.

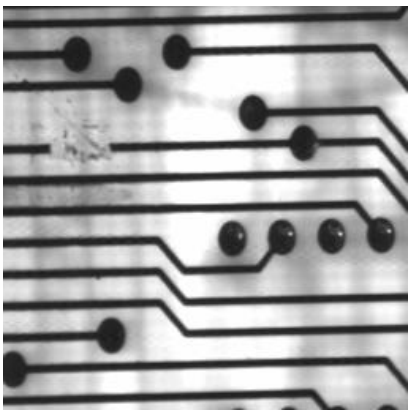
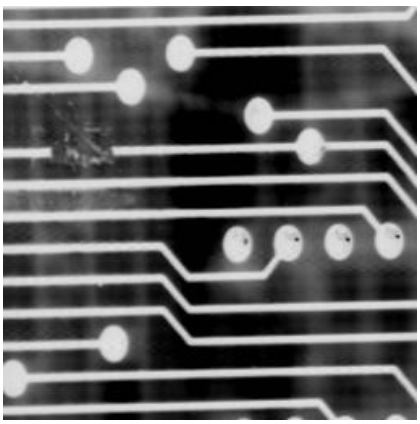
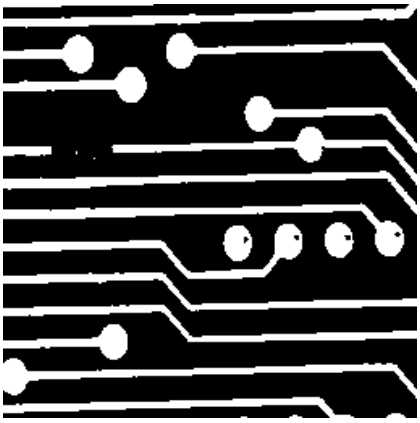


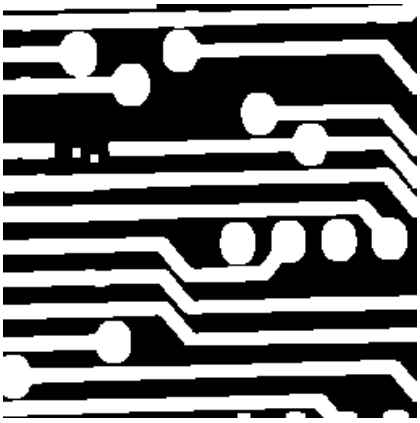
Image originale que l'on souhaite traiter



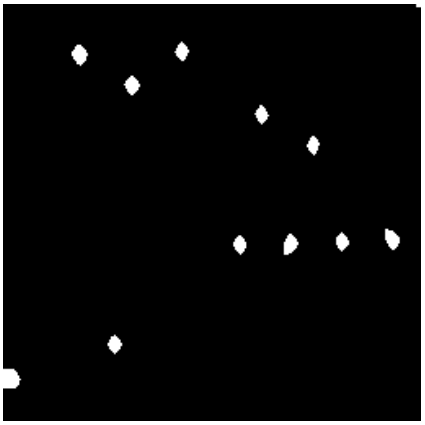
Négatif : Inverse les couleurs de l'image, comme attendu. Du coup, on pense à diviser l'image en deux niveaux de gris directement en utilisant le bloc threshould. Comme le résultat de automatic threshoulding ça donne une partie blanche dans la coin supérieur droit, on utilise le bloc threshould avec le paramètre 160.



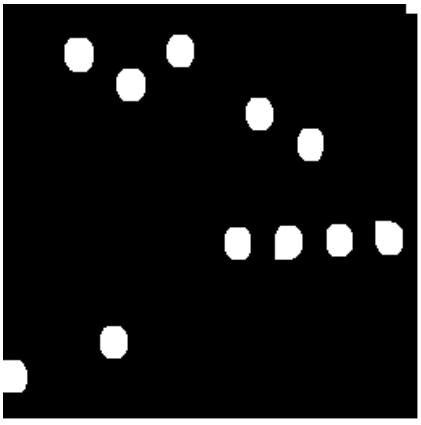
Seuillage : Le filtrage est inutile ici car il est déjà possible de travailler sur l'image. On seuille de manière à séparer les circuits du fond.



Dilatation : Permet de boucher les petits trous à l'intérieur des points nodaux, causés par la réflexion de la lumière. Cela épaissit les points, mais aussi les lignes du circuit.



Érosion en 3 itérations : Permet de se débarrasser des lignes.



Dilatation en 2 itérations : Permet d'épaissir les points comme voulus.