TP1 : Lissage et détection de contours.

Abdelmajid Koubach
Abdellah Belaid

Group 2

March 20, 2022

Abstract

L'objectif du TP est d'étudier 2 techniques de convolution (spatiale dans l'espace image et fréquentielle dans l'espace de Fourrier) soit pour lisser une image (première partie), soit pour effectuer une détection de contours (deuxième partie).

1 Lissage Linéaire

1.1 Filtrage fréquentielle

Le filtrage fréquentiel gaussien est programmé dans la fonction **lissage** du fichier source **filtre_gaussien.c**. Dans les images ci-dessous on a utilisé le filtrage fréquentiel sur l'image **formes2bb40** (à gauche) et on obtient après filtrage de valeur $\sigma = 3$ l'image à droite:

Maintenant, on essaye de déterminer la valeur de σ pour laquelle l'image débruitée est la plus proche à l'image théorique. On trace alors l'évolution du **PSNR** par rapport à σ pour différents types de bruits (Figure 3-10).

1.1.1 Analyse

On remarque que la valeur maximale du **PSNR** des bruits faibles est toujours supérieure à celle des bruits forts. De plus, les valeurs de

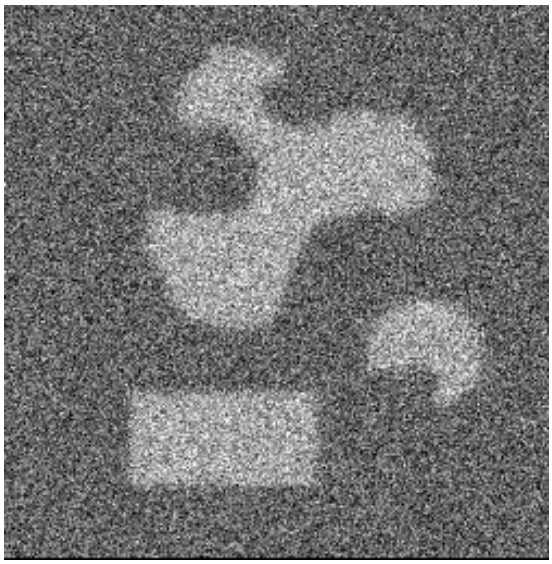

Figure 1: Image avant filtrage.

Figure 2: Image après filtrage (sigma = 3).

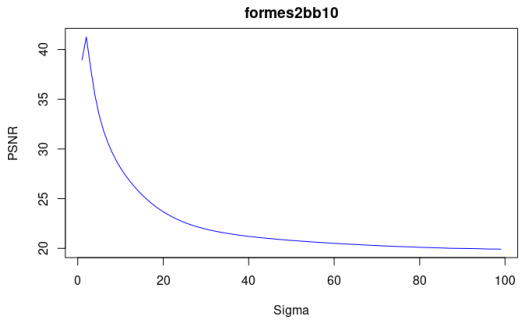

Figure 3: Bruit gaussien faible

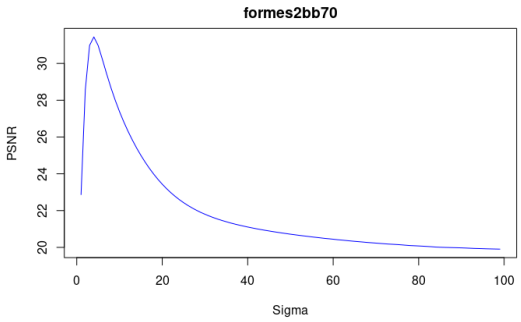

Figure 4: Bruit gaussien fort

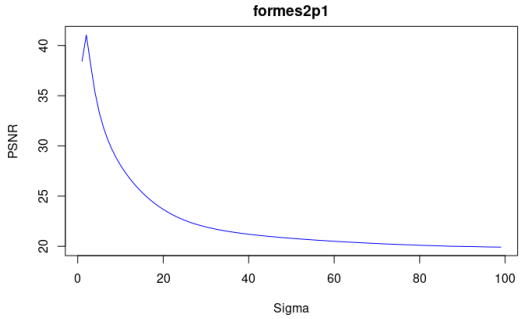

Figure 5: Bruit poisson faible

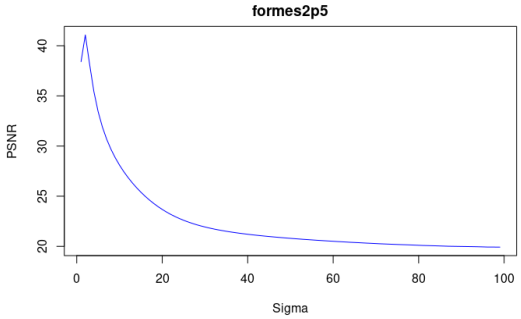

Figure 6: Bruit poisson fort

σ correspondantes des bruits forts sont supérieures à celles des bruits faibles. A l'exception du bruit de type sel et poivre dont le **PSNR** n'est pas très élevée, le filtre gaussien est efficace sur les autres types de bruit.

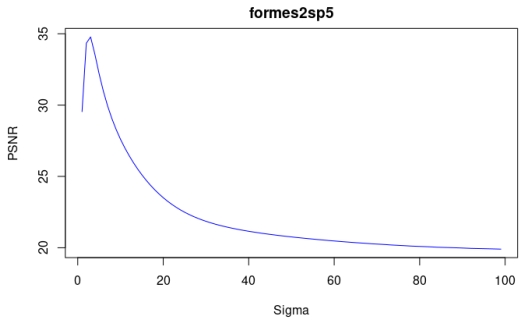

Figure 7: Bruit sel et poivre faible

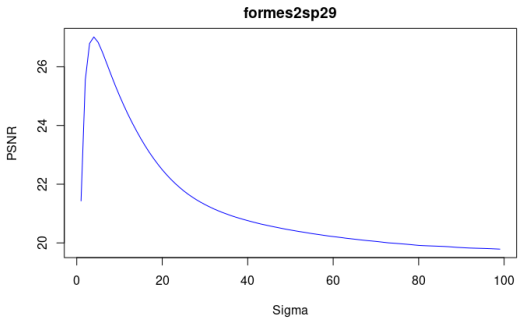

Figure 8: Bruit sel et poivre fort

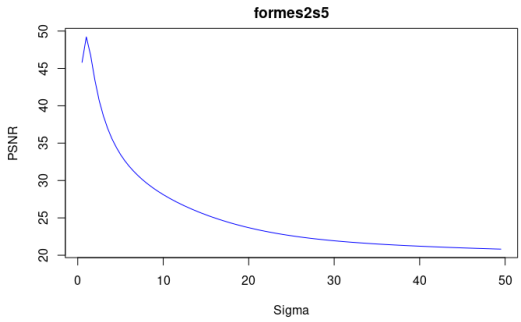

Figure 9: Bruit speckle faible

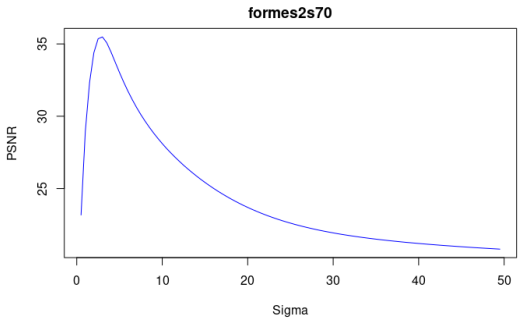

Figure 10: Bruit speckle fort

1.1.2 Résultats du filtre gaussien sur les différents type de bruits forts

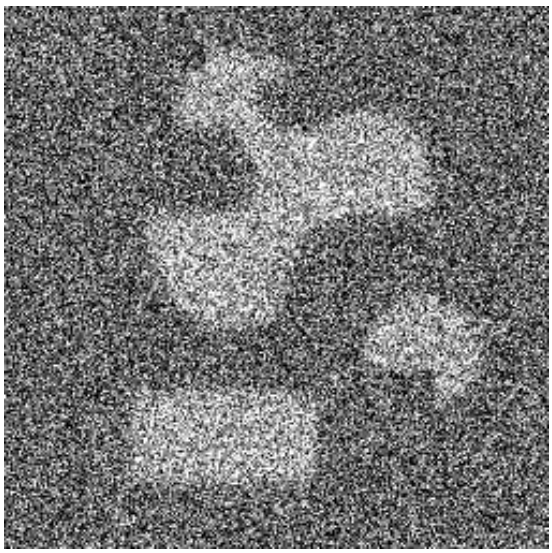

Figure 11: Bruit gaussien avant filtrage

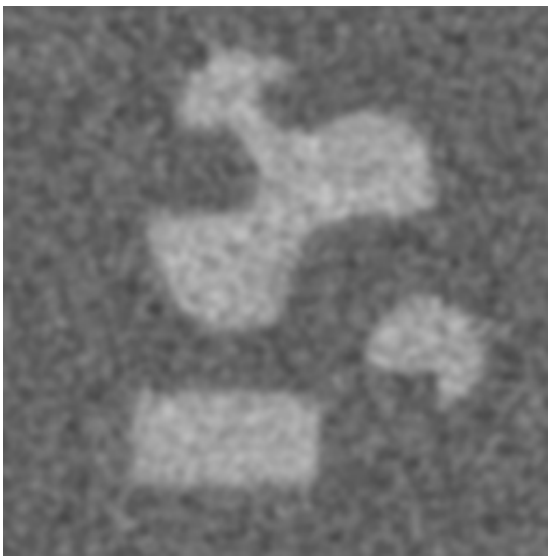

Figure 12: Bruit gaussien après filtrage

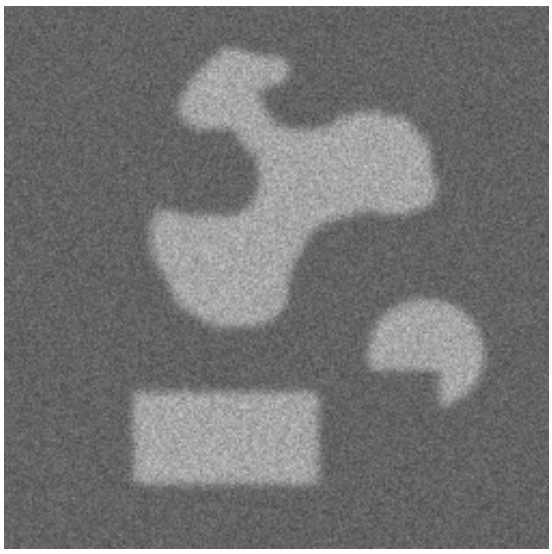

Figure 13: Bruit poisson avant filtrage

Figure 14: Bruit poisson après filtrage

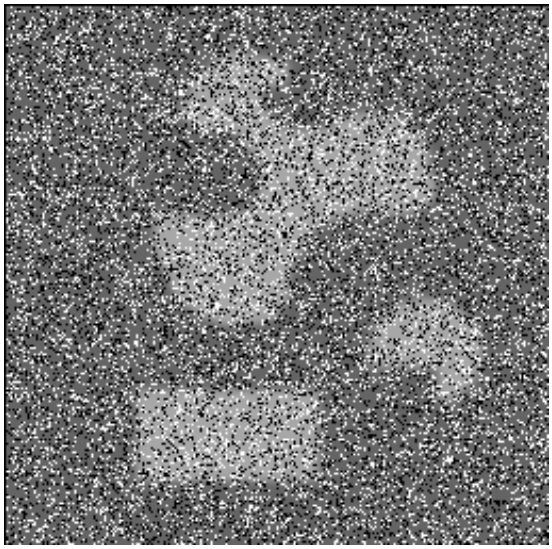

Figure 15: Bruit SP avant filtrage

Figure 16: Bruit SP après filtrage

On conclut d'après les figures 11-18 que le filtrage gaussien n'est pas efficace pour les bruits de type sel et poivre ainsi que les bruits gaussiens qui sont trés forts (2bb70 par exemple).

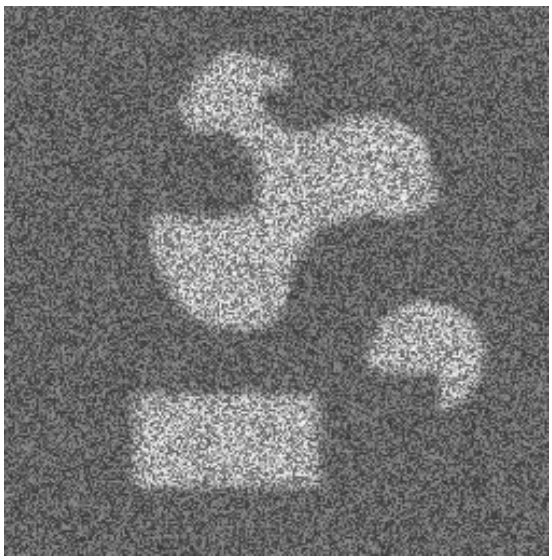

Figure 17: Bruit speckle avant filtrage

Figure 18: Bruit speckle après filtrage

1.2 Convolution spatiale

1.2.1 Implémentation

La convolution spatiale 2D est réalisée dans la fonction **lissage_2D** du fichier source **convolution_2D.c** et la convolution spatiale séparable dans la fonction **lissage_separable** du fichier source **convolution_separable.c**.

1.2.2 EQM : formes2g.pgm et radio1024.pgm

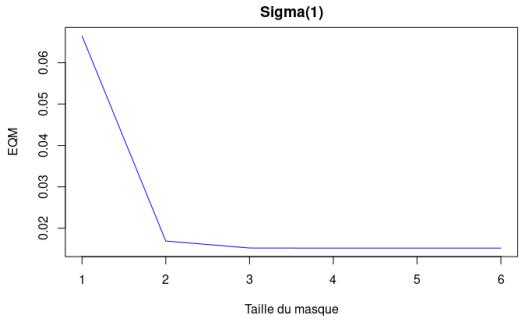

Figure 19: EQM de formes2g pour $\sigma = 1$

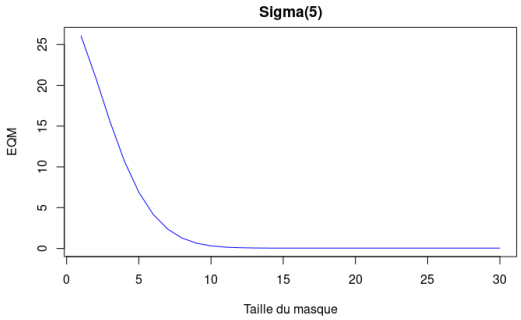

Figure 20: EQM de formes2g pour $\sigma = 5$

D'après les figures 19-26, on remarque que l'écart quadratique moyen est décroissant jusqu'à ce qu'il se stabilise sur une valeur proche de 0 à partir de 3.5σ. Et c'est donc à partir de 3.5σ que l'image produite

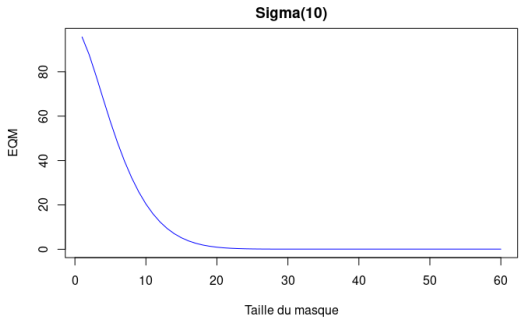

Figure 21: EQM de formes2g pour $\sigma = 10$

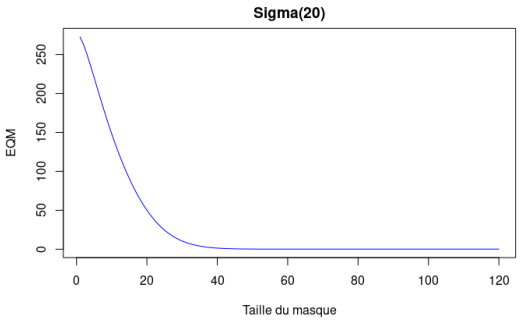

Figure 22: EQM de formes2g pour $\sigma = 20$

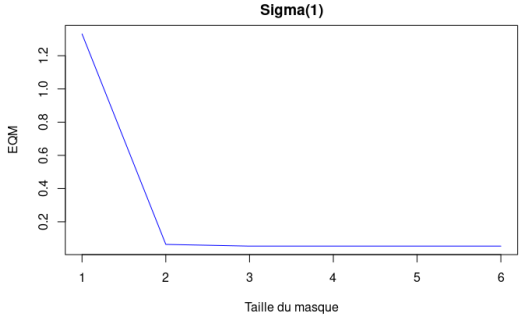

Figure 23: EQM de radio1024 pour $\sigma = 1$

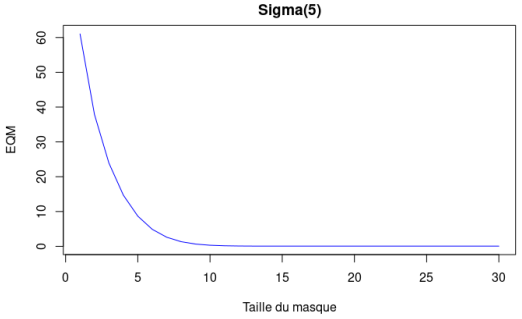

Figure 24: EQM de radio1024 pour $\sigma = 5$

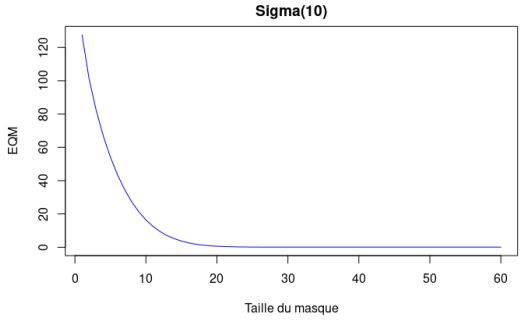

Figure 25: EQM de radio1024 pour $\sigma = 10$

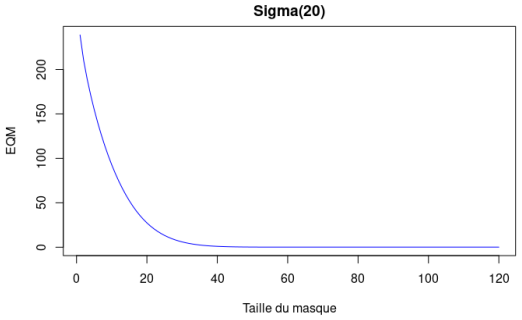

Figure 26: EQM de radio1024 pour $\sigma = 20$

par convolution spatiale devienne identique à celle produite par filtrage fréquentiel.

1.3 Complexité

Cette partie est implémentée dans le fichier **complexite.c** et le traçage s'est fait avec le fichier source **complexite_graph.py**.

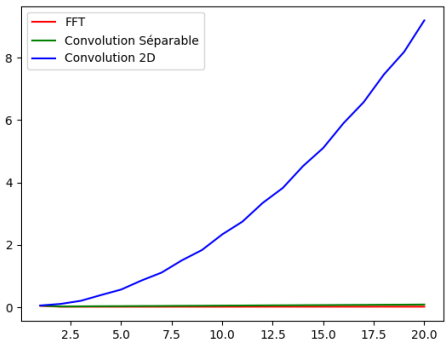

Figure 27: Comparaison des temps des 3 implémentions pour radio1024.pgm

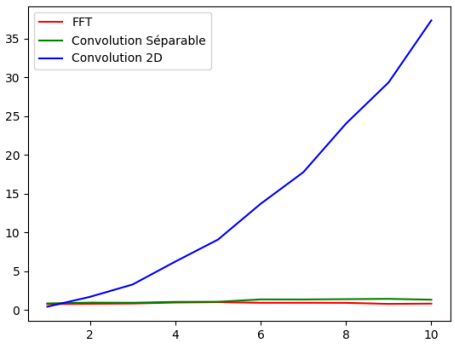

Figure 28: Comparaison des temps des 3 implémentions pour formes2g.pgm

On peut clairement remarquer que la convolution spatiale 2D est très lente par rapport aux deux autres. En effet, la convolution spatiale séparable est la version optimisée de celle-ci de complexité $\mathcal{O}(\mathrm{M}\times\mathrm{N}\times\mathcal{W}(\sigma))$, M et N étant les dimensions de l'image à filtrer. D'autre part la complexité du filtrage fréquentiel est de $\mathcal{O}(\mathrm{M}\times\mathrm{N}\times\ln(\mathrm{M}\times\mathrm{N}))$. On peut en conclure alors que l'utilisation de la convolution spatiale est plus pertinente lorsque la taille de l'image est grande et la valeur σ est petite. En revanche, le filtrage fréquentiel est plus efficace pour des tailles petites M et N.

2 Détection de contours

Les contours peuvent être détectés par un examen des grandeurs étudiées (niveau de gris, distance,couleur, ...) soit :
— par des opérateurs locaux de différentiation du premier ordre.
— par des opérateurs locaux de différentiation du second ordre.
L'objectif de cette partie est de comparer les 2 premières approches qualitativement.

2.1 Opérateur du premier ordre: Gradient

Dans cette partie on a choisit d'implémenter le Gradient discret . cette méthode est réalisée dans le fichier **gradient_discret.c**.

2.1.1 Gradient discret

- Voici un exemple appliqué sur l'image **formes2g.pgm**, en utilisant les filtres de **Prewit**

Figure 29: Image originale bruité

Figure 30: gradient appliqué avec filtre de Prewitt

2.1.2 Extraction des contours

Pour cela on a choisit d'être un peu précis et passer par l'interpolation , mais on a traiter uniquement le cas pour G_x et G_y de même signe ce qui a fallait une perte un d'information spécifiquement pour les lignes parfaitement verticales ou horizontales, avant d'effectuer un seuillage par hysteresis.

Voici un exemple qui illustre ce phénomène:

Or pour des images plus complexes on obtient des résultats satisfaisantes.

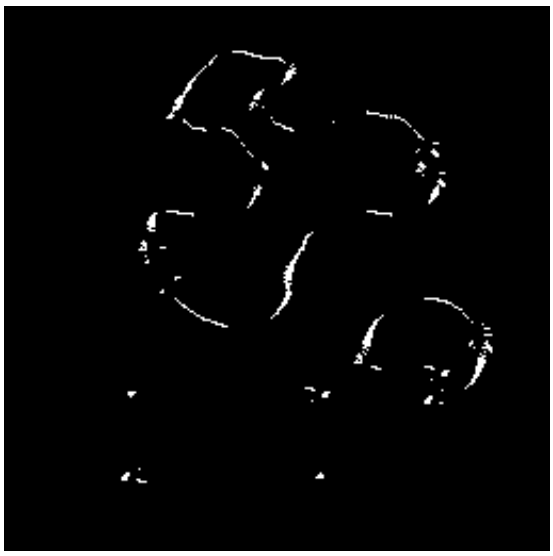

Figure 31: Extraction de contours de la Figure 30

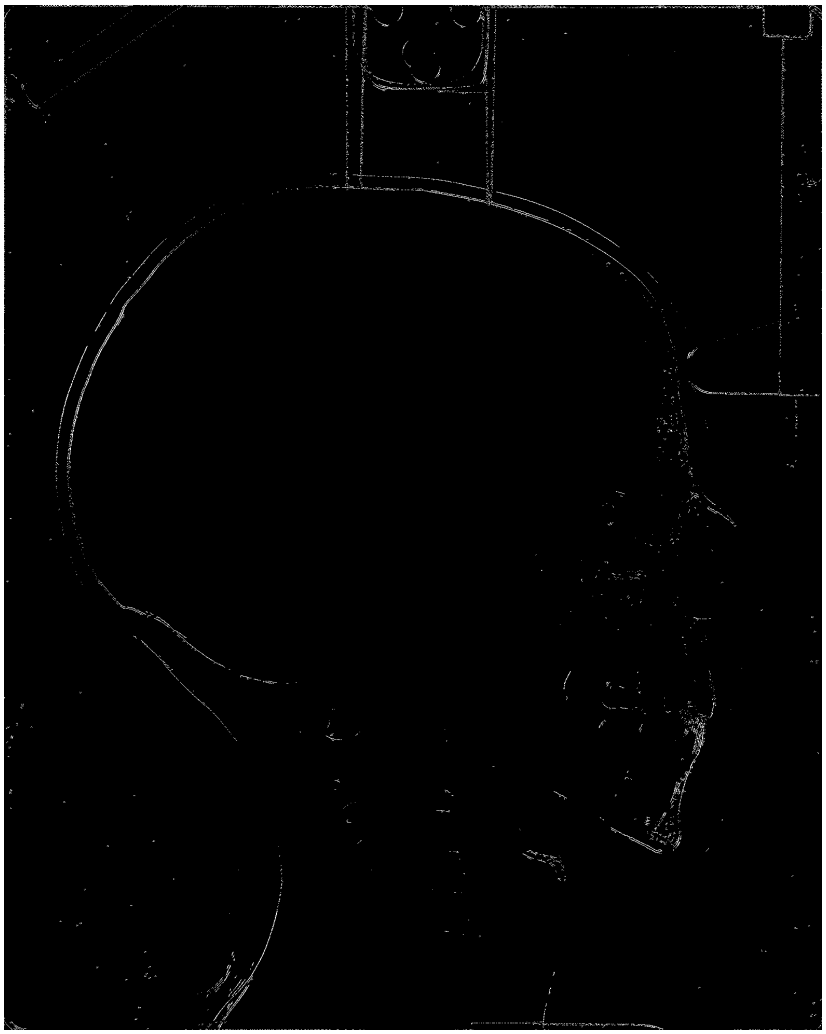

Figure 32: Extraction de contours sur une image bien détaillée.

2.2 Opérateur du deuxième ordre: Laplacien

Contrairement à la partie du gradient on a utilisé la transformée de Fourier **FFT**.

2.2.1 Opérateurs laplacien, lissage et FFT

Comme précédemment on applique notre opérateur sur l'images **formes2.pgm** on obtient:

Figure 33: Image originale bruité

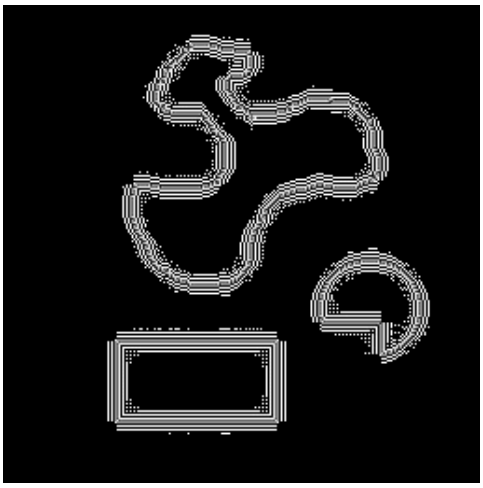

Figure 34: laplacien appliqué avec $\sigma = 0.1$

Figure 35: laplacien appliqué avec $\sigma = 1$

Figure 36: laplacien appliqué avec $\sigma = 10$

On remarque que la qualité du laplacien de l'image dépend de σ ici la figure 35 montre le résultat espérée .

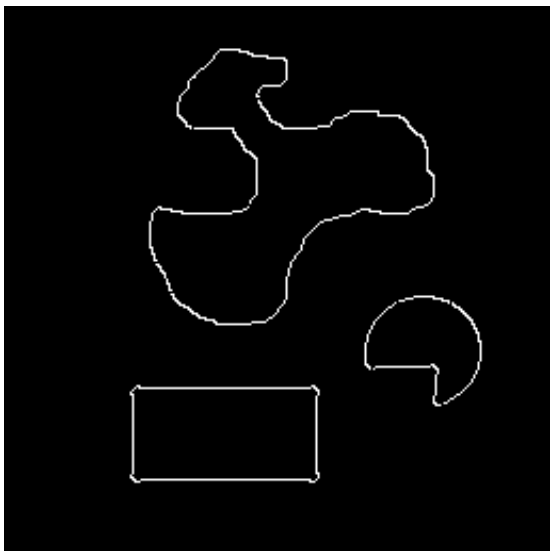

Figure 37: Extraction de contours de la Figure 35

2.2.2 Extraction de contours

Cette fois ici, l'algorithme est très simple , on ajuste un seuil pour un seuillage avec une comparaison avec juste 2 voisin .
On continue sur le même exemple pour $\sigma = 1$ on obtient le résultat:

un autre exemple de l'image **tangram.pgm** en changent le seuil S_h:

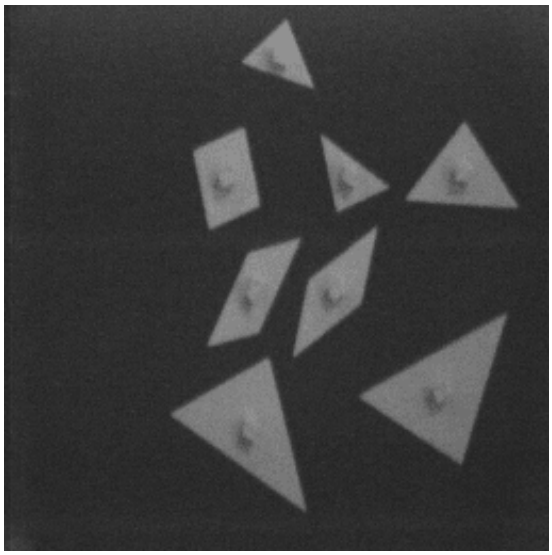

Figure 38: Image originale **tangram.pgm**

Figure 39: Extraction avec seuil $S_h = 0.01$

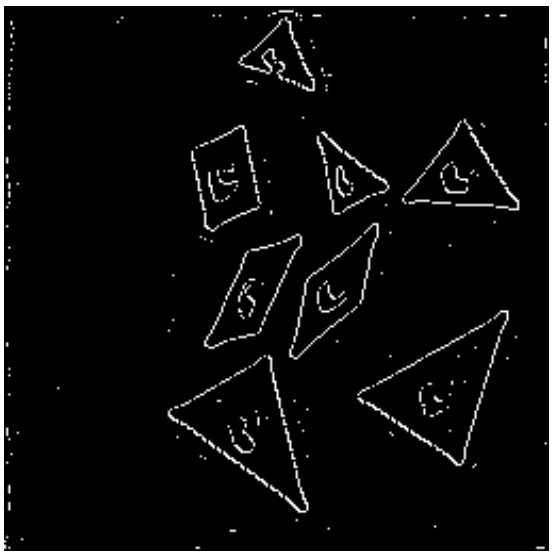

Figure 40: Extraction avec seuil $S_h = 0.2$

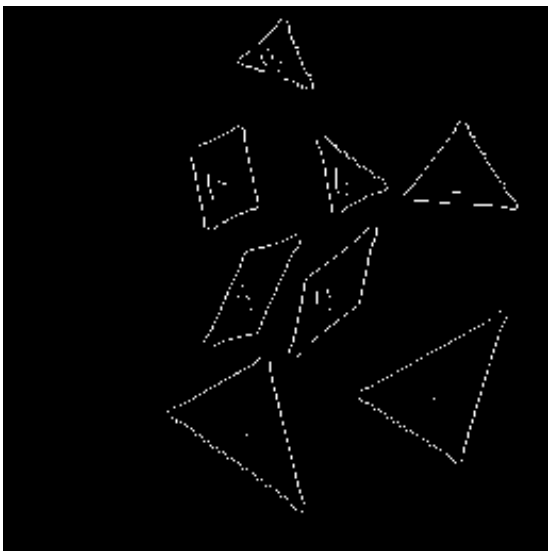

Figure 41: Extraction avec seuil $S_h = 0.9$

On remarque que l'augmentation du seuil donne une image moins bruité avec une détection de contours plus exactes mais à partir d'un certain rang on commence à perdre de l'information et alors on détecte pas des **vrais contours**.

2.2.3 Conclusion

D'un point de vue qualitatif le résultat de l'opérateur laplacien apparaît d'être plus pertinent, ainsi puisqu'on se base sur la **FFT** on a la possibilité de changer *sigma* et l'adapter à notre image pour avoir le meilleur résultat , aussi du point de vue de complexité l'extraction de contours est beaucoup plus simple pour le laplacien que pour le gradient.