Traitement d'images
TP 2

Garcia Maxime, Palisse Benjamin, Petitbon Pierre

L'objectif du TP est de comparer l'efficacité de plusieurs filtres de manière qualitative et quantitative. Le calcul du PSNR entre l'image de départ et l'image filtrée va nous permettre de faire l'étude quantitative des filtres. L'archive contient un Makefile dans le dossier src/ qui a plusieurs fonctionnalités :

-**make compile** qui compile chacun des filtre.

-**make execNomDuFiltre** qui permet d'executer le filtre NomDuFiltre pour toutes les images fournies (les différentes commandes sont execbilat execadapt execmedian execconvolGaussian execconvolGaussianSeparable execgaussian). Attention l'execution du filtre adaptatif et bilatéral peuvent prendre un certain temps. Il faut aussi noter que pour le filtre adaptatif prendre une valeur trop petite pour K (5 ou moins) peut entrainer l'apparition de NaN dans les calculs (cf section filtre adaptatif).

-**make execall** qui exécute tous les filtres pour toutes les images.

-**make clean**.

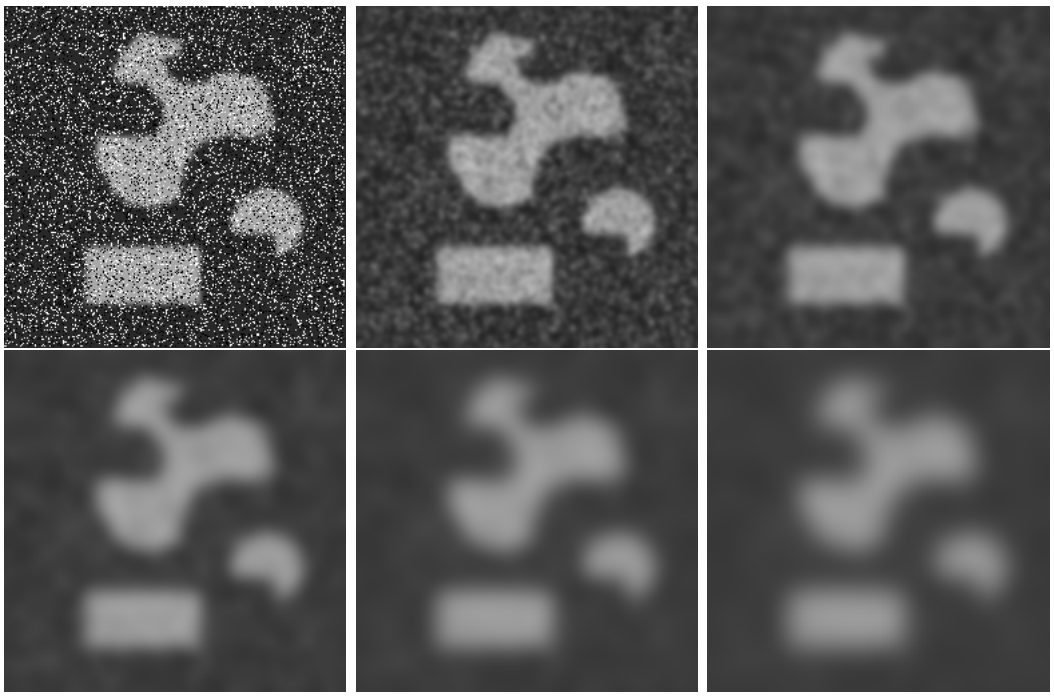
1 Filtre Gaussien

Nous avons traité ce filtre de trois manières différentes : en utilisant la FFT, en procédant directement avec une convolution spatiale discrète et en réalisant deux convolutions spaciales 1D en utilisant le caractère séparable du filtre. L'intérêt d'utiliser la transformée de fourier dans le cas de la gaussienne (de formule $G(x,y) = \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{x^2+y^2}{2\sigma^2})$) est double car on peut montrer que $TF(G(u,v)) = \exp(-2\pi^2\sigma^2(u^2+v^2))$ (nous n'avons donc pas besoin d'utiliser l'algorithme de la FFT sur la gaussienne) et le coût du calcul de l'image filtrée est en $\Theta(nlog(n))$ avec n le nombre de pixels. Dans le cas d'une convolution spatiale discrète le coût est de $\Theta(nm)$ avec m la taille du masque. Il est donc intéressant d'utiliser la FFT lorsque m $\ggg$ log(n). Comme la gaussienne est une fonction C^∞ à décroissance rapide la convolution avec l'image de départ va régulariser cette dernière, c'est pour cela que les images de sorties sont plus lisses que l'image de départ (effet de flou). On observe par ailleurs que plus σ est grand plus l'image est floutée. Ceci s'explique par le fait que

l'on pondère la valeur d'un nouveau pixel par celle de son voisinage et plus la taille σ est grande plus le voisinage considéré est grand.

1.1 FFT

Voici quelques résultats pour différentes tailles σ (formes1pets5 originale puis pour $\sigma =$ 2,4,6,8,10) :



On constate bien sur cet exemple que plus σ augmente plus l'image est floutée mais cet effet est plus important sur le bruit que sur les formes (ce qui est normal car une moyenne sur un voisinage d'un bruit va transformer ce dernier plus efficacement que sur les formes ou le voisinage varie peu). On peut donc noter une certaine efficacité à effacer le bruit même s'il persiste encore et que les formes s'en retrouvent aussi déteriorées ($\sigma = 2$ donne un résultat assez satisfaisant).

Voici le PSNR associé aux différents exemples :

	$\sigma = 2$	$\sigma = 4$	$\sigma = 6$	$\sigma = 8$	$\sigma = 10$
gaussien	38.24	30.116290	26.442085	24.206949	22.618504
speeckle	22.91	22.094201	21.243987	20.429643	19.685113
poivre et sel	11.72	11.482070	11.396079	11.324898	11.256465

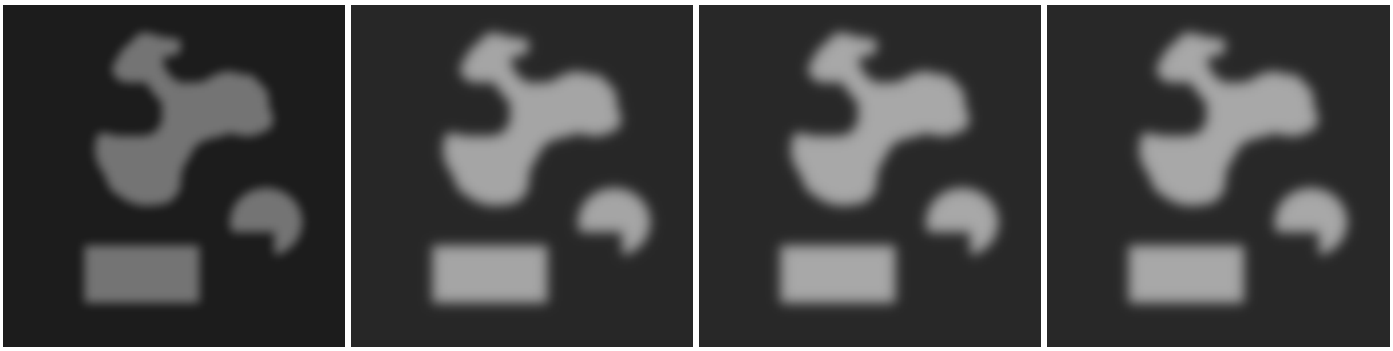
Le PSNR a un comportement cohérent car il décroit lorsque σ augmente ce qui est logique puisque plus ce dernier est grand plus l'image est floutée et plus l'image filtrée est éloignée

de l'image de départ.

1.2 Convolution spaciale séparée

Une autre manière d'appliquer le filtre gaussien est de passer directement par la convolution spatiale. Ceci rend le calcul plus long ($\Theta(nm)$ avec m la taille du masque)) mais a l'avantage de pouvoir mieux contrôler la taille du voisinage sélectioné pour l'application du masque (c'est le paramètre noté w ($= \sqrt{m}$) qui représente la largeur (ou la longueur) du filtre). On peut remarquer que le filtre gaussien est séparable c'est à dire que $G(x, y) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{x^2}{2\sigma^2}) * \exp(-\frac{y^2}{2\sigma^2}) = u(x) * v(y)$, on peut donc réduire la convolution 2D à deux convolutions 1D en faisant un filtrage horizontal puis un filtrage vertical (l'image a été périodisée pour garder un voisinage uniforme pour les pixels du bord de l'image). Le coût de l'opération est $\Theta(2 * (2 * w * n)) = \Theta(4wn) = \Theta(wn)$.

Pour $\sigma = 4$ (formes1g pour w = 5,10,15,20):



	$w = 5$	$w = 10$	$w = 15$	$w = 20$
gaussien	19.641360	29.896616	30.115347	30.116289
speeckle	17.937598	22.058604	22.094057	22.094201
poivre et sel	10.836366	11.479171	11.482063	11.482070

On remarque que les PSNR sont identiques à ceux de la FFT lorsque $w >= 20$

Pour $\sigma = 8$ (formes1sp pour w=5,10,15,20):

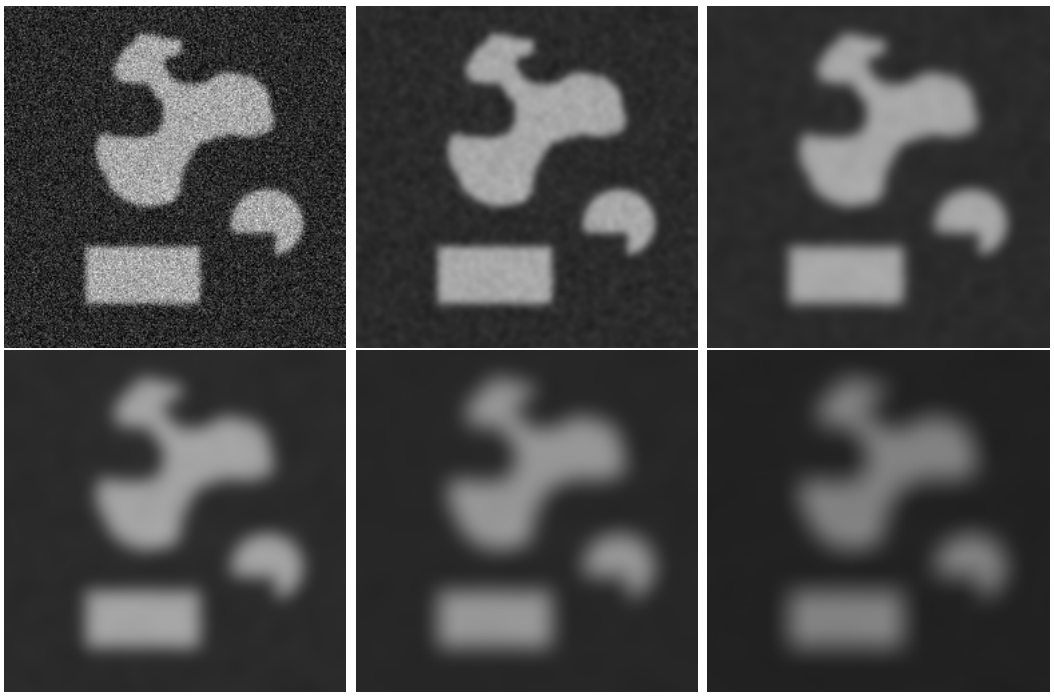
	$w = 5$	$w = 10$	$w = 15$	$w = 20$
gaussien	12.337654	18.127763	23.067826	24.129553
speeckle	11.957846	16.848061	19.915799	20.396852
poivre et sel	8.597150	10.585879	11.250320	11.321084

pour $\sigma = 8$ les PSNR sont identiques à ceux de la FFT lorsque $w >= 40$

Si on fixe $w = 15$ et que l'on fait varier σ :

	$\sigma = 2$	$\sigma = 4$	$\sigma = 6$	$\sigma = 8$	$\sigma = 10$
gaussien	38.237998	30.115347	26.316742	23.067826	19.805811
speeckle	22.910568	22.094057	21.205957	19.915799	18.030303
poivre et sel	11.716086	11.482063	11.392473	11.250320	10.924719

formes1bb25 puis image filtrée avec σ = 2,4,6,8,10 :



On sait que pour le filtre gaussien lorsque $w = 3 * \sigma$ la courbe est décrite à 99.5%.C'est pour cela que le PSNR varie peu et que le rendu est quasi identique lorsque w dépasse cette valeur. On retrouve par ailleurs l'effet de lissage décrit précédemment lorsque σ augmente (à w fixé) mis à part que la décroissance du PSNR est plus rapide car w est fixe et que la courbe n'est plus décrite à 100% lorsque σ augmente. On remarque encore qualitativement, comme pour la FFT, que le filtrage gaussien est assez efficace pour masquer le bruit même si les formes s'en retrouvent un peu détériorées.

1.3 Convolution spaciale

Faire une convolution spaciale en 2D ne change rien aux résultats obtenus précédemment, seul le coût de l'algorithme change. Celui-ci est en $\Theta(w^2 n)$ et non plus en $\Theta(wn)$. On

peut maintenant comparer le temps que met chaque algorithme pour s'exécuter sur une image de dimension 256*256 (formes1bb10) :

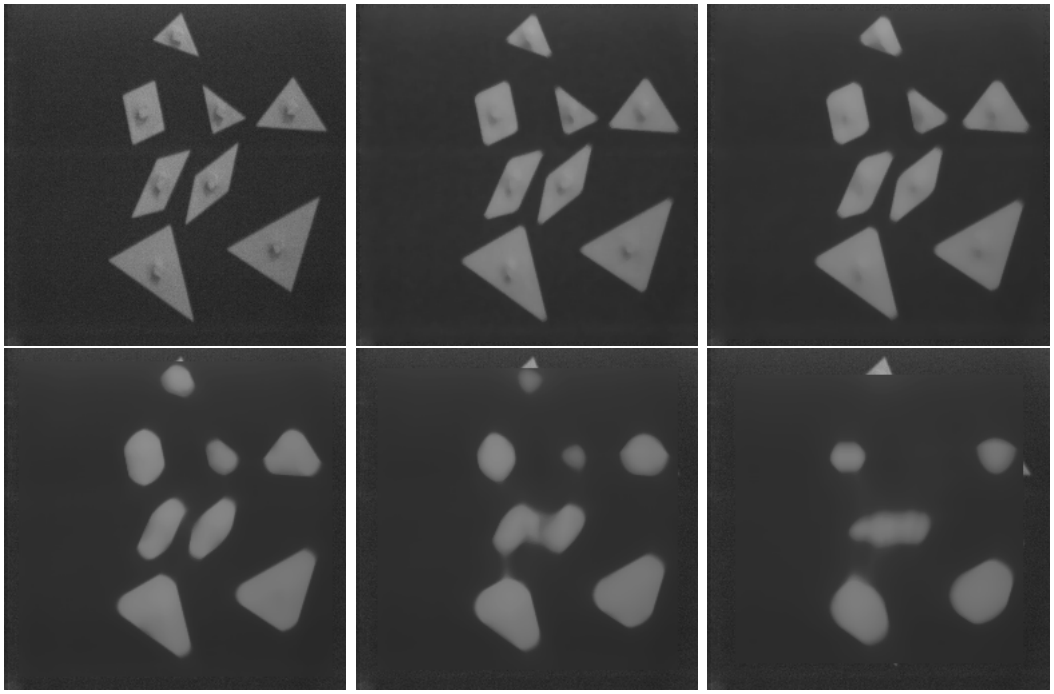
	$w = 5$	$w = 10$	$w = 16 = log(n)$
FFT	0.084s	0.084s	0.084s
Convolution séparée	0.156s	0.251s	0.367s
Convolution 2D	2.088s	6.934s	16.738s

On remarque que dans tous les cas la FFT est plus rapide que la convolution séparée ce qui peut paraître contradictoire avec ce que nous vu plus haut au niveau de la complexité mais il ne faut pas oublier les facteurs de multiplicité du coût qui entre en jeu ici. Par ailleurs on peut voir que la convolution séparable est beaucoup plus rapide que la convolution 2D (les facteurs de multiplicité étant très proches pour ces deux filtres) ce qui correspond à nos attentes.

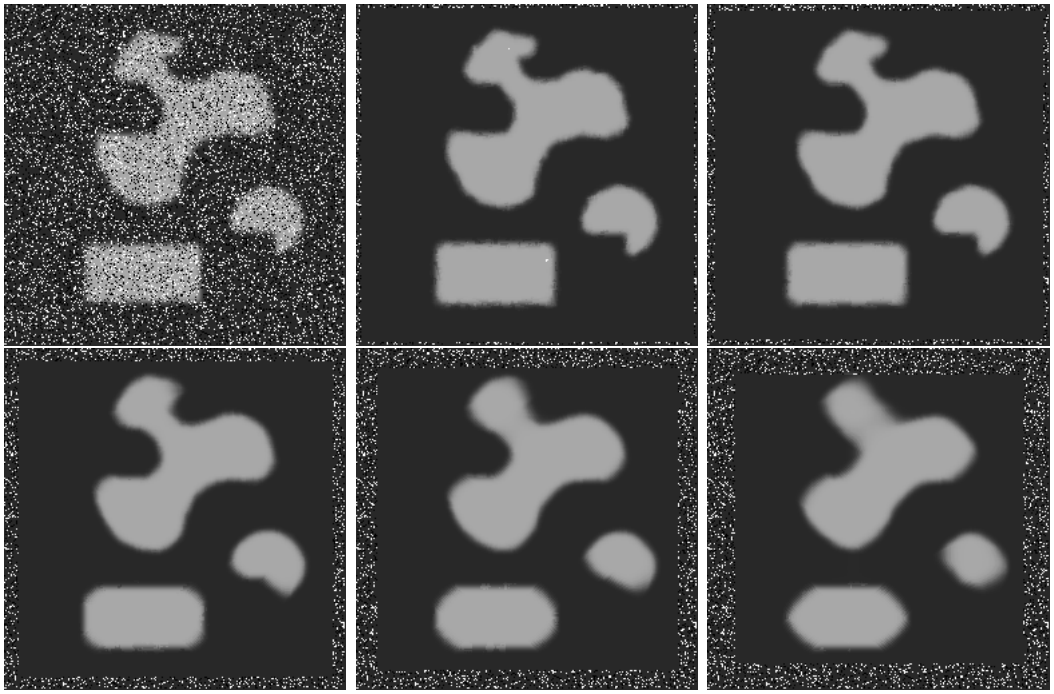
2 Filtre Median

Contrairement au filtre gaussien le filtre médian n'effectue pas une pondération d'un pixel par son voisinage mais lui attribut la valeur médiane de ce dernier. Ce filtre permet donc de bien corriger les imperfections isolées et il n'introduit pas de nouvelle intensité puisque que l'on attribut à chaque pixel une valeur déjà existante. Cependant les contours des formes creuses auront tendance à disparaitre après passage du filtre car ils seront considérés comme des imperfections isolées.
lacornou puis image filtrée pour N = 3,5,10,15,20 :

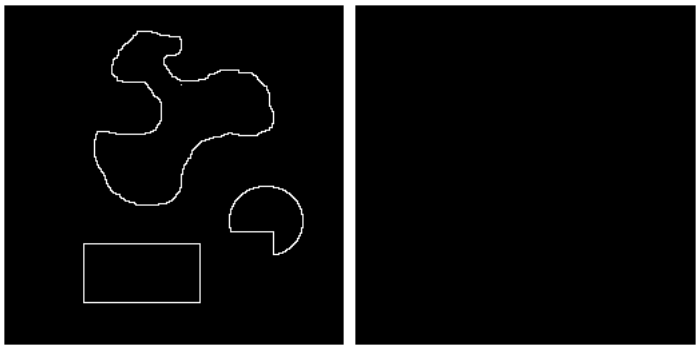
tangram puis image filtrée pour N = 3,5,10,15,20 :

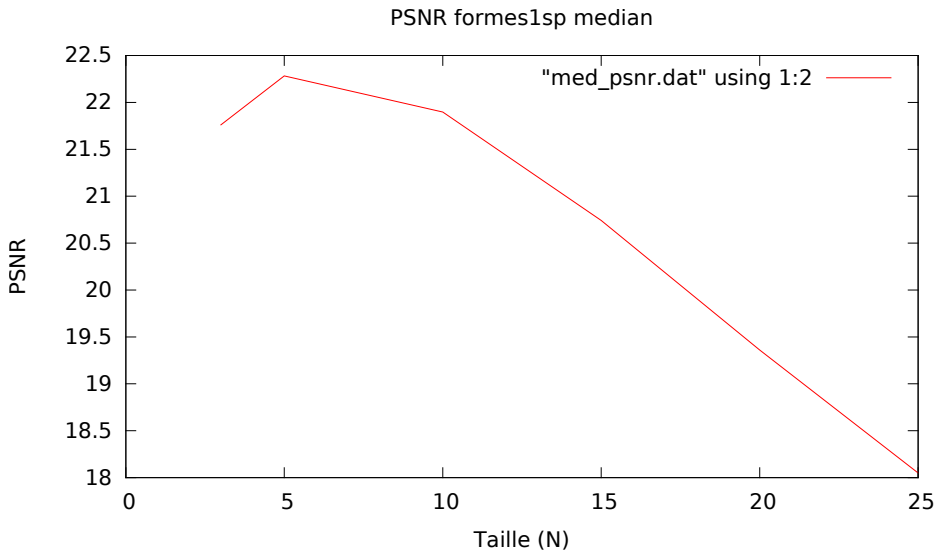


formes1pets5 puis image filtrée pour N = 3,5,10,15,20 :



formes1c1 puis image filtrée pour N = 3 :



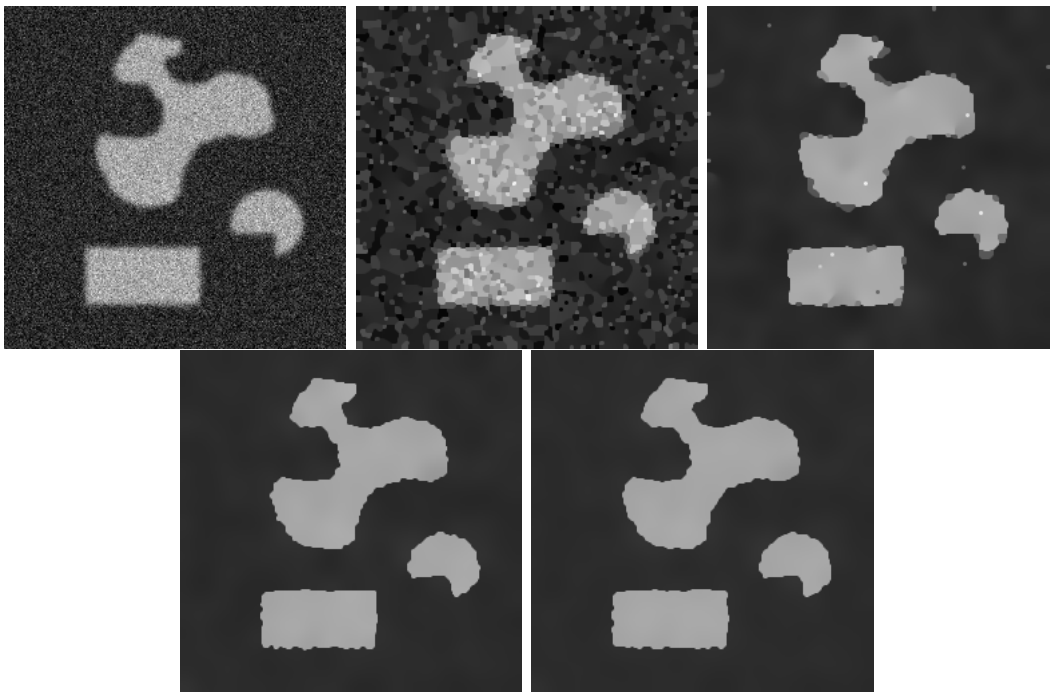


On peut noter que l'algorithme ne traite pas les bords de l'image, c'est pour cela que l'on retrouve l'image originale sur une bordure de la taille du filtre. Pour tangram et lacornou qui sont deux images régulières et sans bruit, on observe que les contours sont d'autant plus rognés que la taille du filtre est grande. L'intérieur des bords à tendance à s'homogénéiser. Néanmoins on constate la grande efficacité du filtre à effacer le bruit avec l'image poivre et sel, ce dernier est totalement effacé à l'intérieur de la fenêtre contrairement au filtrage gaussien ou le bruit était encore présent. Pour formes1c1 on observe bien le phénomène décrit précédemment à savoir que les contours de formes creuses disparaissent après passage du filtre. On constate que le psnr diminue lorsque la taille du filtre augmente, ce qui s'explique par le fait que plus N augmente plus les déformations sont importantes (puisque la médiane aura tendance à changer).

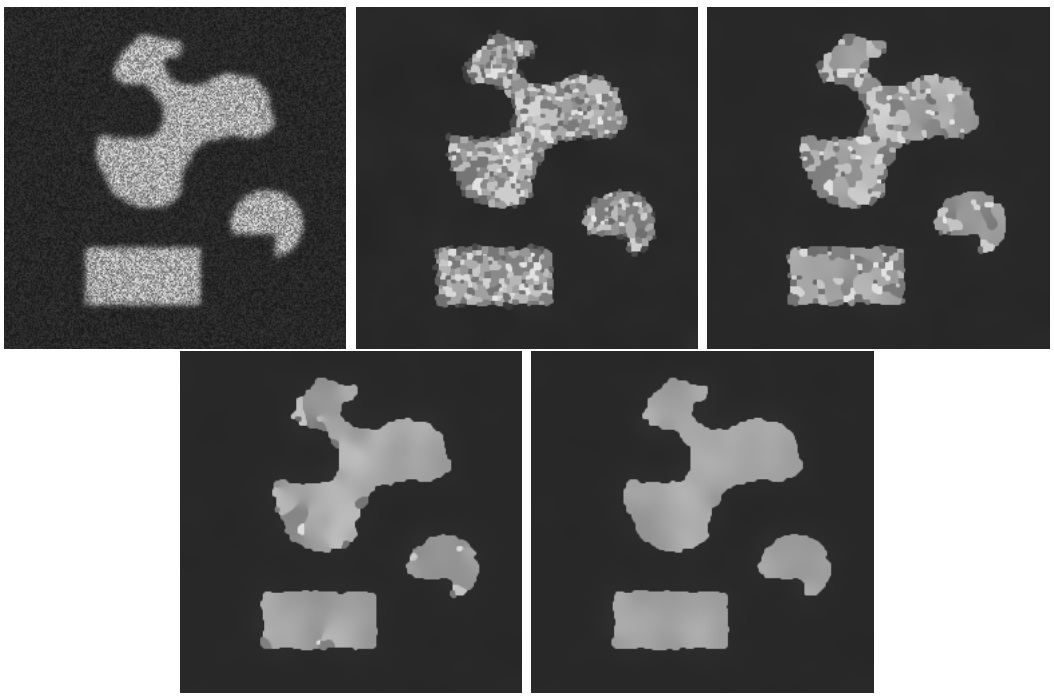
3 Filtre Adaptatif

Le filtre adaptatif a pour but de palier au problème rencontré pour le filtre médian à savoir le filtrage des contours. Le filtrage adaptatif agit fortement sur les zones homogènes et faiblement sur les zones à gradient élevé, de cette manière ci on peut espérer

que les contours seront conservés tout en éliminant le bruit. Il faut noter que lorsque $(G_x^t)^2 + (G_y^t)^2$ est trop grand (c'est le cas d'une image bruitée avec du bruit blanc car 2 pixels voisins ont une valeur totalement différente), il faut que k soit aussi assez grand pour que $\exp(-\frac{(G_x^t)^2+(G_y^t)^2}{2k^2})$ ne soit pas approximé par 0 (ce qui ferait apparaitre des NaN dans le calcul et l'image de sortie serait noire).
formes1bb25 puis image filtrée pour k = 5,10,15,20 :

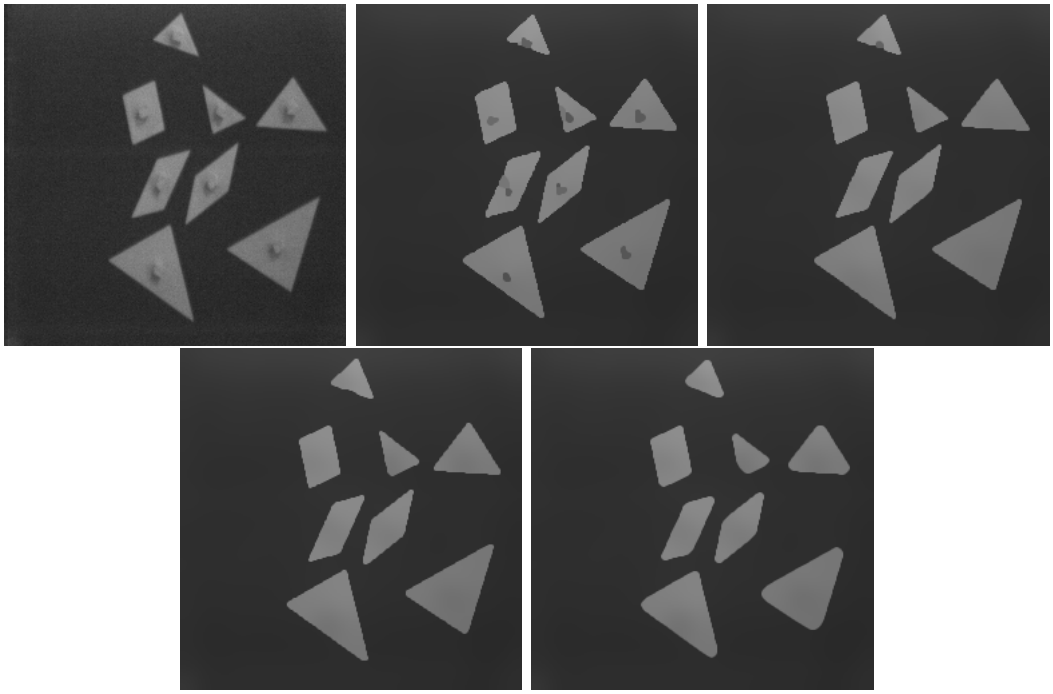


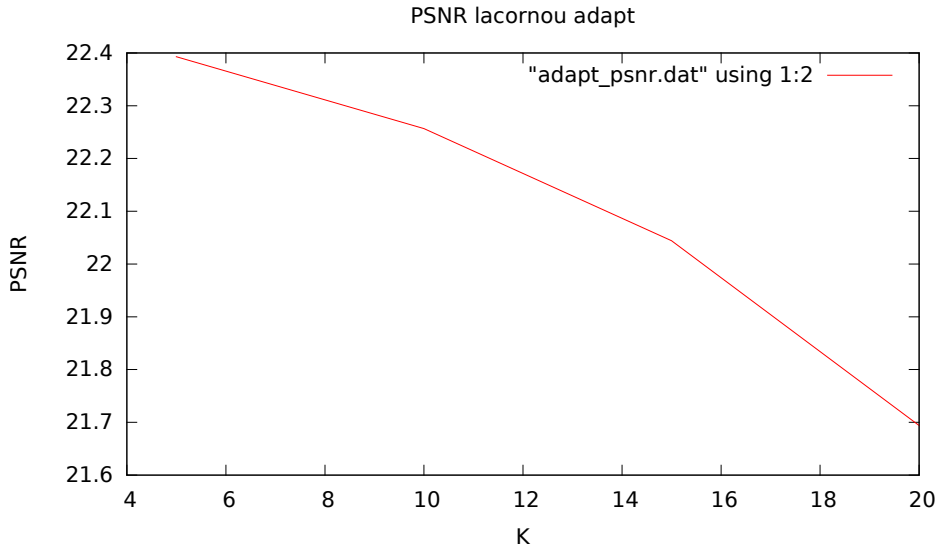
formes1sp puis image filtrée pour k = 5,10,15,20 :



lacornou puis image filtrée pour k = 5,10,15,20 :

tangram puis image filtrée pour k = 5,10,15,20 :

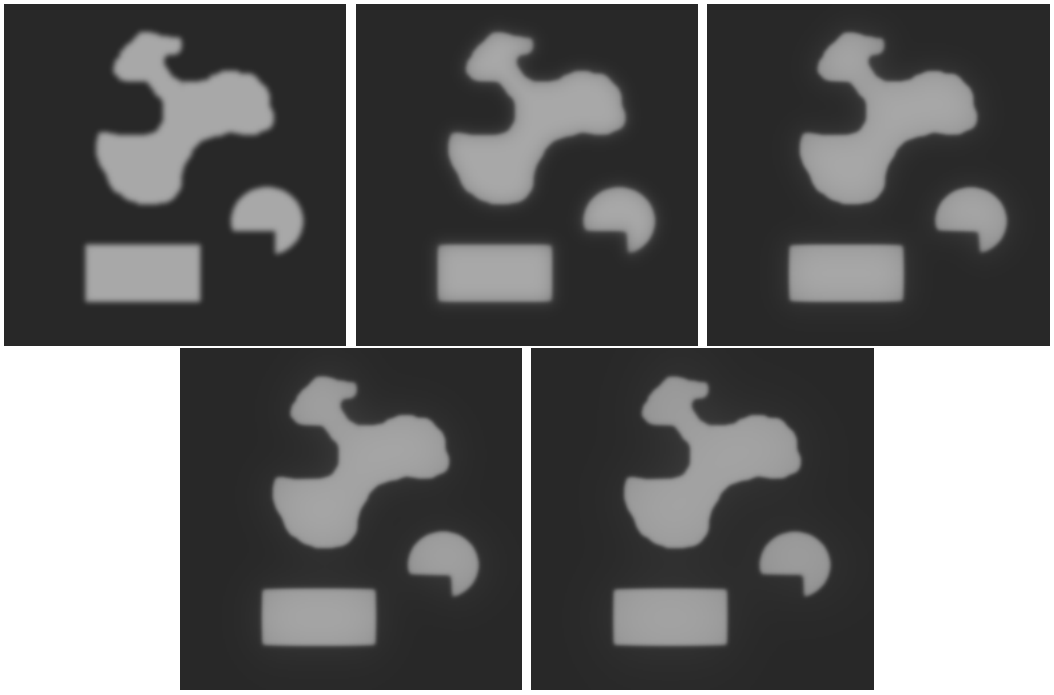




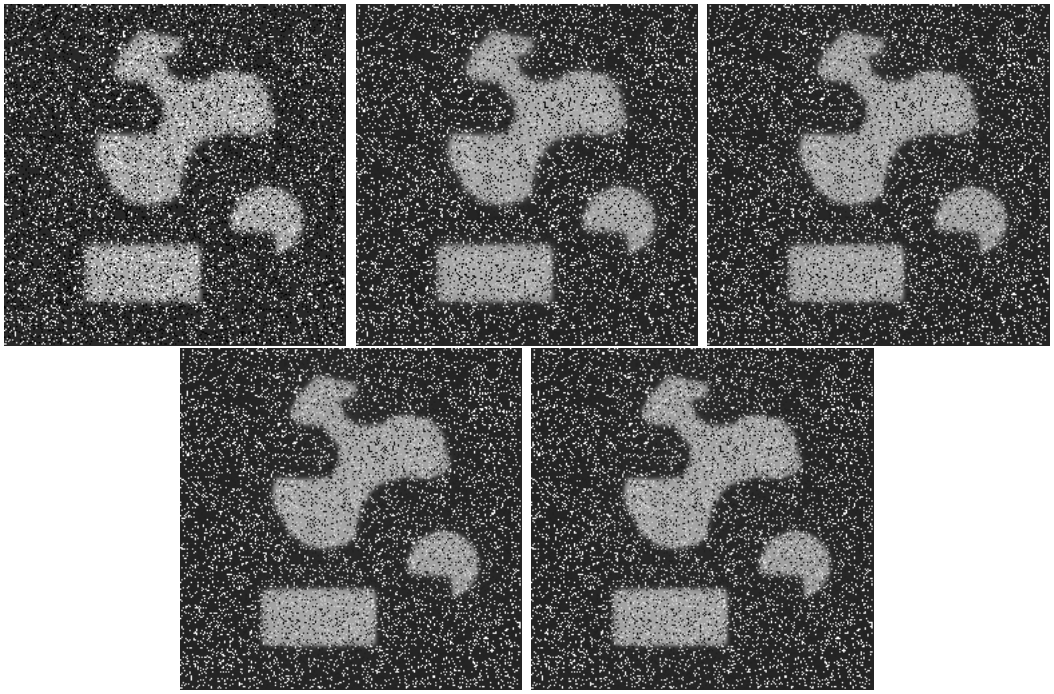
On observe que le filtre adaptatif est un bon débruiteur pour un bruit blanc (et pour le speckle) et qu'il conserve bien les formes homogènes. Malheureusement comme on peut le voir sur lacornou et tangram il efface tous niveaux de gris intermédiaire et transforme l'image d'origine en image contenant de grande zones avec le même niveau de gris. Nous perdons donc tous les détails de l'image d'origine. De plus, comme pour le filtre médian, nous observons cet effet de rognement des contours de l'image lorsque k augmente. Néanmoins on observe bien que le filtre conserve bien les contours (et même celui des formes creuses), comme on peut le voir sur lacornou ou les mats sont toujours présents après passage du filtre (contrairement au filtre médian). On constate que le psnr diminue losrque k augmente ce qui est cohérent avec le fait que plus il est grand moins de zones homogènes de même niveau de gris il y a (mais elles sont plus grandes). Pour des images de bruit blanc le psnr reste à peu près constant, ceci peut s'expliquer par le fait que les images sont constituées de très peu de niveaux de gris.

4 Filtre Bilatéral

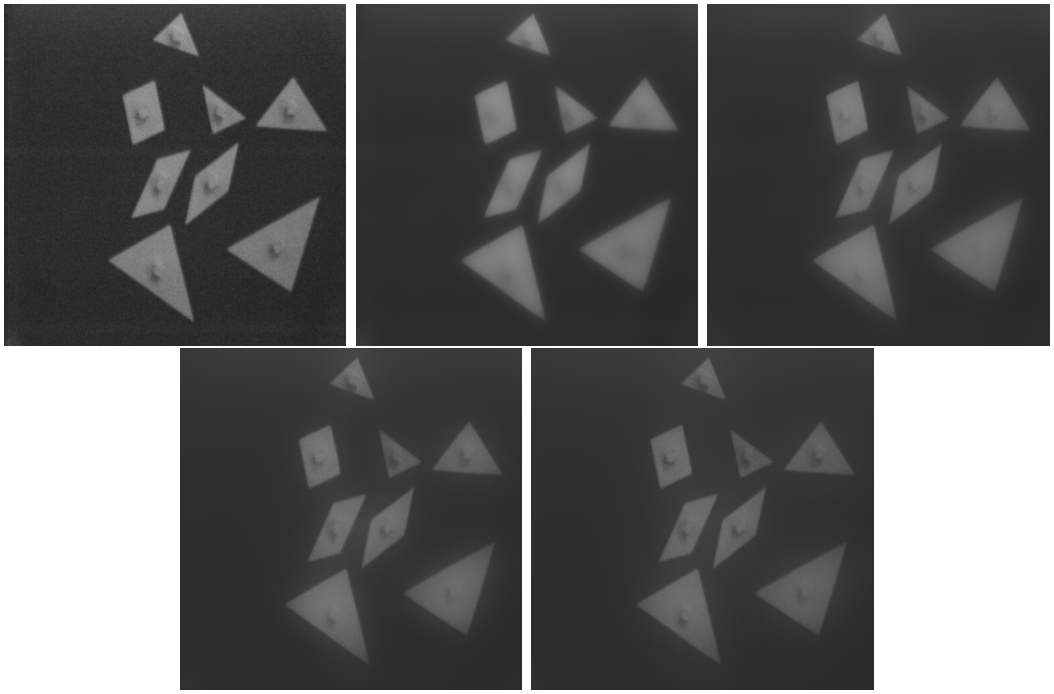
Le filtre bilatéral se base sur le même principe que le filtre gaussien, c'est à dire que chaque pixel est pondéré par la valeur de ses voisin, mais il considère en plus la différence des intensités du pixel central et du pixel du masque.
formes1g puis image filtrée pour sigma1 = 5,10,15,20 :



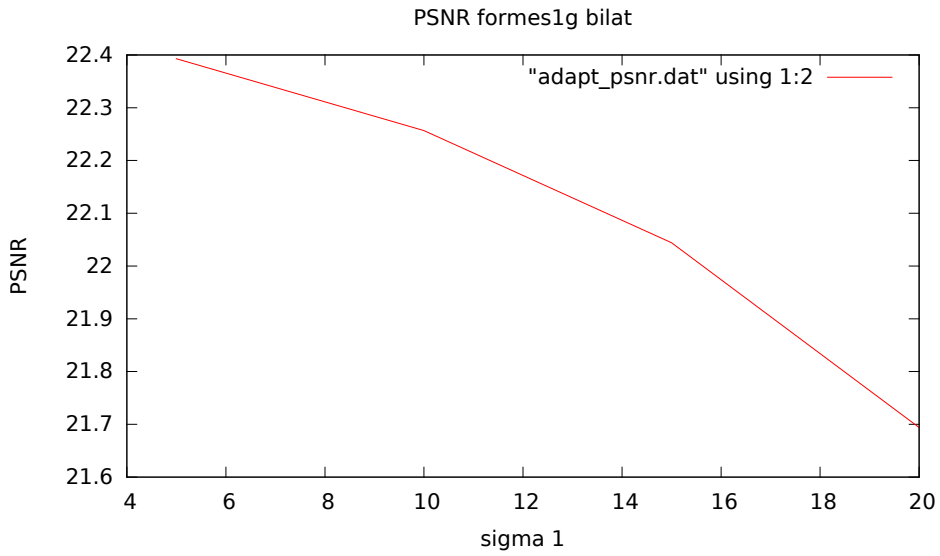
formes1pets5 puis image filtrée pour sigma1 = 5,10,15,20 :



tangram puis image filtrée pour sigma1 = 5,10,15,20 :



lacornou puis image filtrée pour sigma1 = 5,10,15,20 :



On constate qualitativement que le filtre bilatéral a peu d'effet sur les images, il homogénéise un peu ces dernières (c'est pour cela qu'il a tendance à débruiter un petit peu les images comme tangram et formes1sp) en introduisant un effet de flou. Néamoins on voit clairement sur poivre et sel que ce filtre n'est pas adapté au débruitage. Le psnr diminue à cause de l'effet de flou provoqué par le filtre.