Vision par Ordinateur :
Filtrage d'Images

Objectif du Filtrage d'Images

Le filtrage d'images est une technique utilisée en vision par ordinateur pour améliorer la qualité d'une image, supprimer les bruits et extraire des caractéristiques importantes. En termes simples, c'est comme passer une image à travers un tamis pour enlever les impuretés et rendre l'image plus claire.

Exemples :

- *Améliorer la netteté d'une photo floue.*
- *Réduire le bruit dans une image prise dans des conditions de faible luminosité.*
- *Détecter les contours des objets dans une image.*

Sources de Dégradation

Les images peuvent être dégradées pour plusieurs raisons, y compris :

- **Bruit** : Il s'agit de variations aléatoires de couleur ou de luminosité dans une image. Le bruit peut être causé par des capteurs de caméra de mauvaise qualité, une faible luminosité ou des interférences électroniques.

- **Flou :** Le flou peut se produire lorsque l'appareil photo bouge pendant la prise de la photo ou lorsque l'objet est en mouvement.

- **Distorsion :** Les lentilles des caméras peuvent parfois provoquer des distorsions, comme les lignes droites qui apparaissent courbées.

Exemple d’images bruitées

Bruit d'acquisition, de numérisation, de transmission

Rendu

Bruit spatial fixe

Bruit de compression

Bruit sous Matlab

J= imnoise(I,'gaussian')	Ajouter un bruit blanc gaussien de moyenne zéro avec une variance de 0,01 à l'image I en niveaux de gris
J= imnoise(I,'gaussian',m)	Ajouter un bruit blanc gaussien avec une moyenne m et une variance de 0,01
J= imnoise(I,'gaussian',m,var_gauss)	Ajouter un bruit blanc gaussien avec une moyenne m et une variance var_gauss
J= imnoise(I,'poisson')	Génèrer un bruit de Poisson à partir des données
J= imnoise(I,'salt & pepper')	Ajouter du bruit sel et poivre, avec une densité de bruit par défaut de 0,05. Cela affecte environ 5% des pixels
J= imnoise(I,'salt & pepper',d)	Ajouter du bruit sel et poivre, où d est la densité de bruit
J= imnoise(I,'speckle')	Ajouter du bruit multiplicatif en utilisant l'équation J = I + n * I, où n est un bruit aléatoire uniformément distribué avec une moyenne de 0 et une variance de 0,05
J= imnoise(I,'speckle',var_speckle)	Ajouter un bruit multiplicatif avec une variance var_speckle

Bruit sous Matlab

Code en Matlab:

```
I=imread('cameraman.tif');
IB=imnoise(I, 'salt & pepper');
imshow(IB);
```

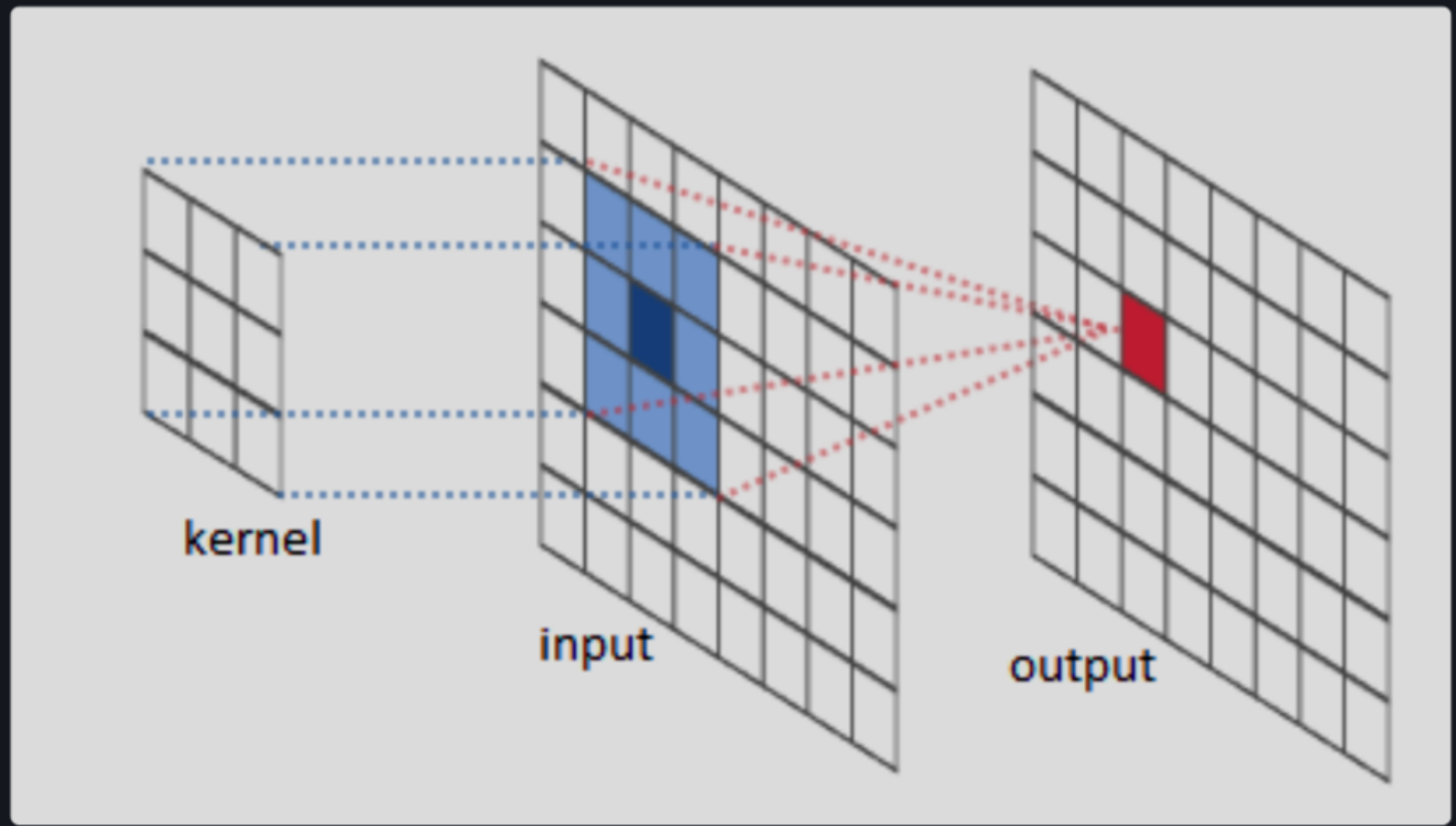
Image originale

Image bruitée

Notion de la Convolution

- La convolution est une opération mathématique qui combine deux ensembles d'informations. En traitement d'image, la convolution est utilisée pour appliquer des filtres à une image. Un filtre est une matrice (ou noyau) que l'on fait glisser sur l'image pour transformer ses pixels.

Convolution d'images : En traitement d'images, une matrice de convolution ou noyau est une petite matrice utilisée pour le floutage, l'amélioration de la netteté de l'image, le gaufrage, la détection de contours, et d'autres. Tout cela est accompli en faisant une convolution entre le noyau et l'image.

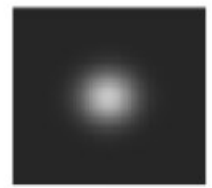


Transformation locale: utilisation du voisinage de chaque pixel

Image originale I

Filtre de convolution f

*



=

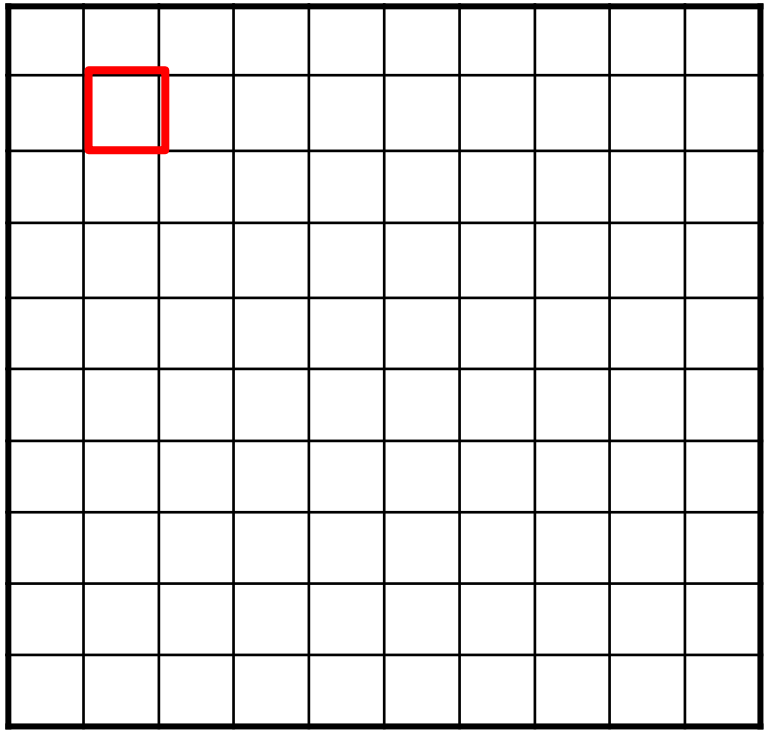
Image convoluée h

Convolution en pratique (1)

$I(.,.)$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h(.,.)$



$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f(\cdot,\cdot)$

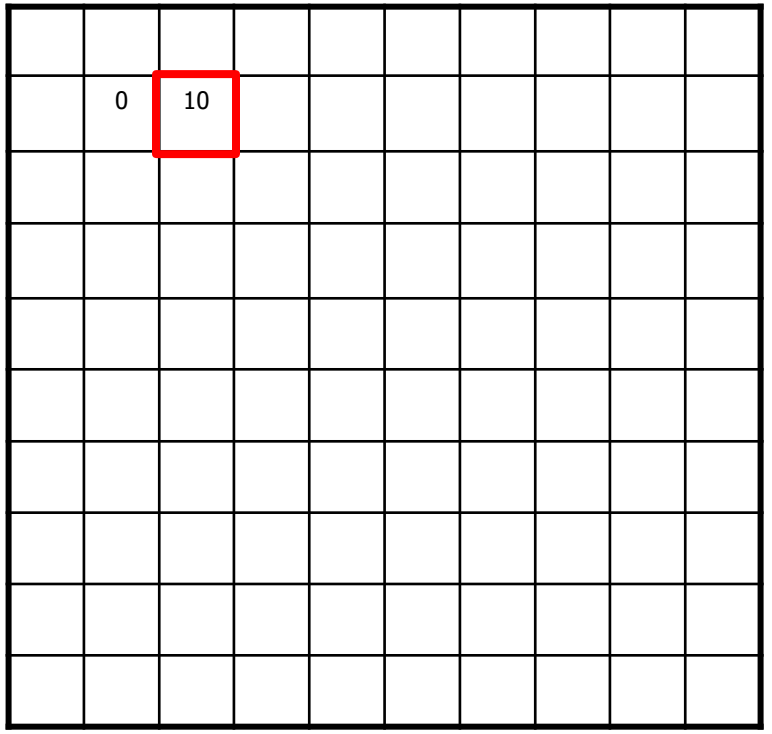
$$h(m,n) = \sum_{k,l} f(k,l) I(m+k, n+l)$$

Convolution en pratique (2)

$I(.,.)$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h(.,$



$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f(\cdot,\cdot)$

$$h(m,n) = \sum_{k,l} f(k,l) I(m+k, n+l)$$

Convolution en pratique (3)

$I(.,.)$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h(.,.)$

	0	10	20						

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f(\cdot,\cdot)$

$$h(m,n) = \sum_{k,l} f(k,l) I(m+k, n+l)$$

Convolution en pratique (4)

$I(.,.)$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h(.,.)$

	0	10	20	30					

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$f(\cdot,\cdot)$

$$h(m,n) = \sum_{k,l} f(k,l) I(m+k, n+l)$$

Convolution en pratique (5)

$I(.,.)$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h(.,.)$

	0	10	20	30	30				

$$\frac{1}{9}\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

$f(\cdot,\cdot)$

$$h(m,n) = \sum_{k,l} f(k,l) I(m+k, n+l)$$

Convolution en pratique (6)

$I(.,.)$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h(.,.)$

	0	10	20	30	30				
				?					

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f(\cdot,\cdot)$

$$h(m,n) = \sum_{k,l} f(k,l) I(m+k, n+l)$$

Convolution en pratique (7)

$I(.,.)$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h(.,.)$

	0	10	20	30	30				
						?			
				50					

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f(\cdot,\cdot)$

$$h(m,n) = \sum_{k,l} f(k,l) I(m+k, n+l)$$

Convolution en pratique (8)

$I(.,.)$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h(.,.)$

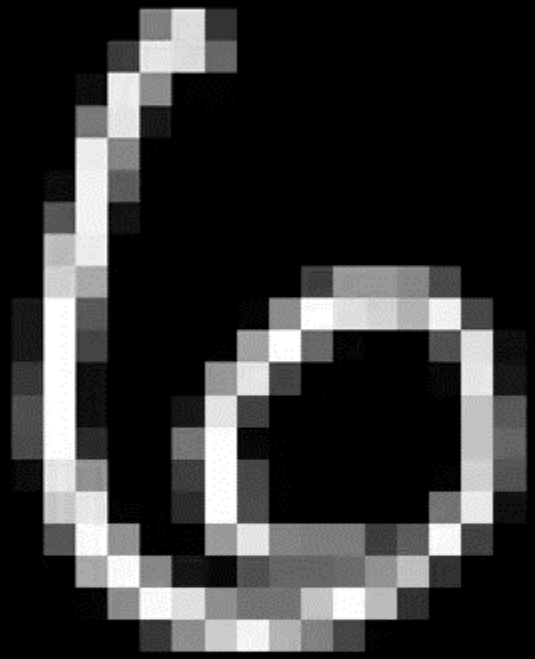
	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f(\cdot,\cdot)$

$$h(m,n) = \sum_{k,l} f(k,l) I(m+k, n+l)$$



Filtres Moyenneurs et Gaussiens

Filtres Moyenneurs :

- Utilisent la moyenne des pixels voisins pour lisser l'image.
- Réduisent le bruit mais peuvent aussi flouter les bords.

Filtre Moyenneur Matlab

Code en Matlab:

```
image= imread('cameraman_bruite.tif');
f=fspecial('average',3);
Image_F = imfilter(image,f);
```

Image originale

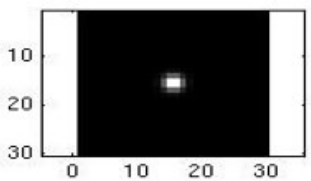
Image bruitée

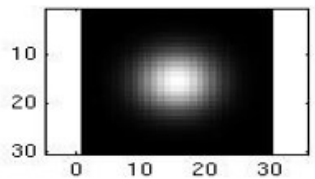
Image filtrée

Filtres Gaussiens :

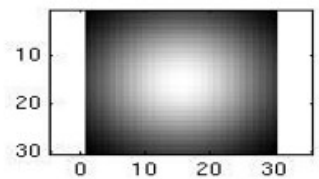
- Utilisent une courbe gaussienne pour un lissage plus naturel.
- Conservent mieux les détails et les bords tout en réduisant le bruit.

Effet du lissage avec différente gaussienne





...



Le Filtre Médian

- Le filtre médian est particulièrement efficace pour supprimer le bruit de type "sel et poivre". Il remplace chaque pixel par la valeur médiane de ses voisins, ce qui préserve mieux les détails par rapport à la moyenne.

Filtre Médian sous Matlab

J = medfilt2(I,[m n])
Avec [m n] le voisinage

Image bruitée

Image filtrée

Code en matlab:

```
J = medfilt2(IB,[3 3])
Imshow(J)
```

Conclusion

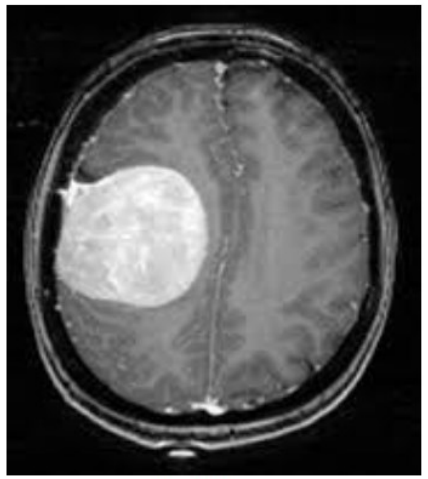
- Le filtrage d'images, en utilisant des techniques comme la convolution, permet de transformer les images pour améliorer leur qualité, extraire des informations précieuses et détecter des motifs spécifiques. C'est une technique fondamentale en traitement d'images, utilisée dans de nombreuses applications pratiques, de la photographie à la vision par ordinateur.

Passons au TP

Le but de ce TP est d’utiliser des fonctions dédiées au filtrage d’image de Matlab pour segmenter les images. Une étude comparative des résultats est préconisée.

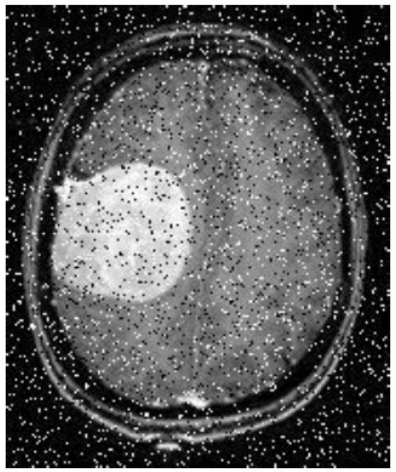
Filtrage passe bas

- On dispose de l'image originale « Ref.jpg » bruitée avec un bruit gaussien (« m_br1.jpg » et un bruit Sel & Poivre (« m_br2.jpg »)
- On souhaite filtrer les images bruitées par des filtres passe bas moyenneur et médian avec différentes tailles de noyaux de convolution (3x3, 7x7 et 13x13)



Ref.jpg

m_br1.jpg



m_br2.jpg

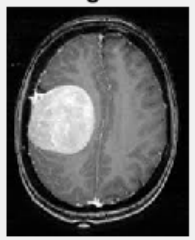
Le travail demandé :

- Sur une première figure, affichez l'image originale et les 3 images filtrées par le filtre moyenneur avec les différentes dimensions du noyau.
- Sur la même figure, affichez les 4 histogrammes des images ; Commentez vos résultats.
- Sur une deuxième figure, affichez les résultats pour le filtre médian en utilisant les différentes dimensions du noyau.
- Expliquez l'effet global des 2 filtres sur l'image, notamment par rapport à la dimension du noyau.

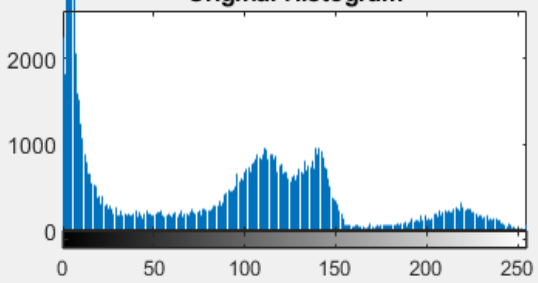
Sous MATLAB, utilisez les fonctions suivantes :

- **fspecial** pour choisir le type de filtre et sa dimension
- **imfilter** pour appliquer un filtre dont il faut préciser ses valeurs.
- **medfilt2** pour appliquer le filtre médian.

Original

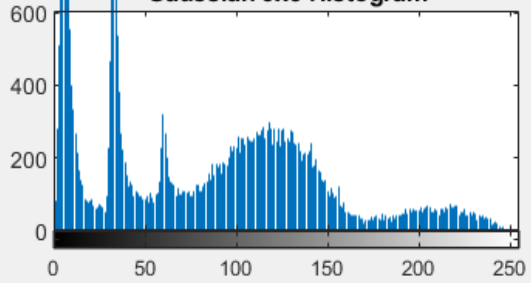


Original Histogram

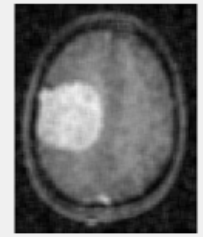


Gaussian 3x3

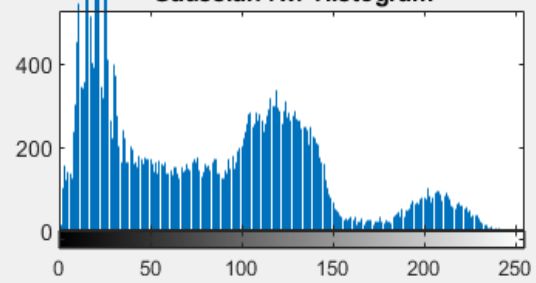
Gaussian 3x3 Histogram



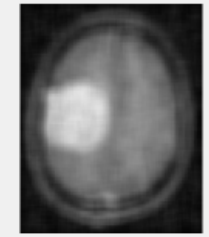
Gaussian 7x7



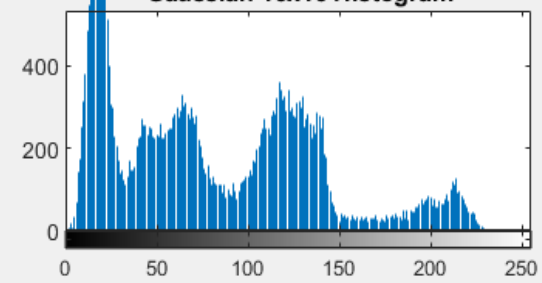
Gaussian 7x7 Histogram



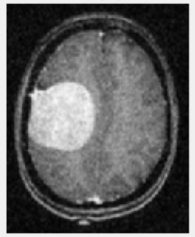
Gaussian 13x13



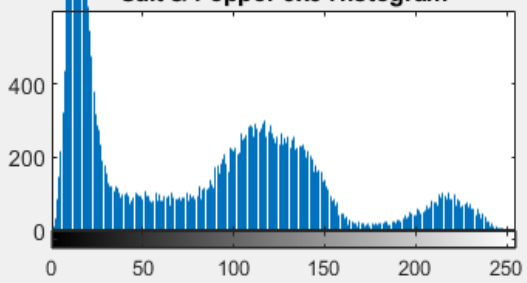
Gaussian 13x13 Histogram



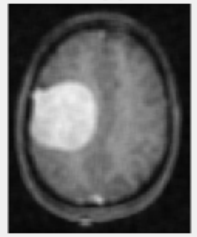
Salt & Pepper 3x3



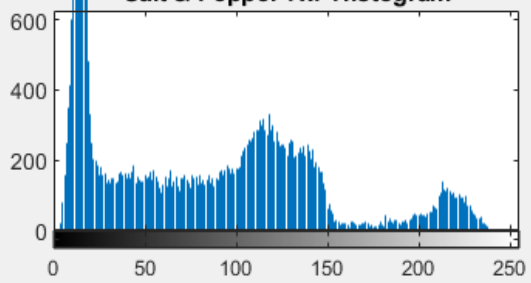
Salt & Pepper 3x3 Histogram



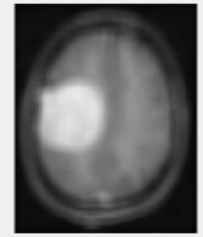
Salt & Pepper 7x7



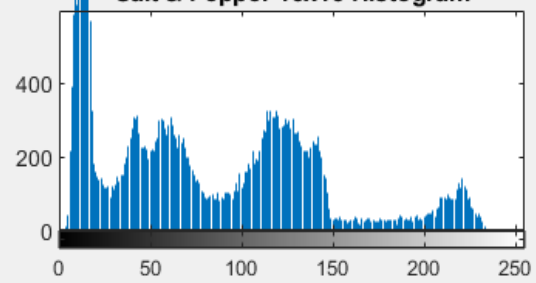
Salt & Pepper 7x7 Histogram



Salt & Pepper 13x13



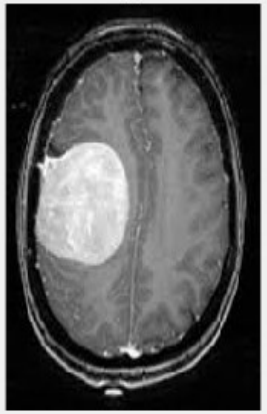
Salt & Pepper 13x13 Histogram



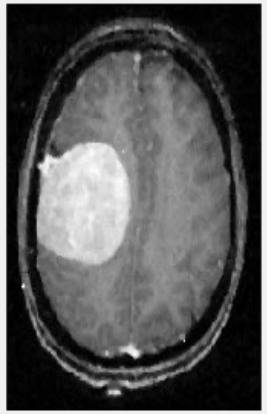
```
% Charger les images
original = imread('ref.jpg');
noisy_gaussian = imread('m_br11.jpg');
noisy_sp = imread('m_br22.jpg');

% Créer les filtres moyenneur avec différentes tailles de noyaux
h3 = fspecial('average', [3 3]);
h7 = fspecial('average', [7 7]);
h13 = fspecial('average', [13 13]);

% Appliquer les filtres sur les images bruitées
filtered_gaussian_3 = imfilter(noisy_gaussian, h3);
filtered_gaussian_7 = imfilter(noisy_gaussian, h7);
filtered_gaussian_13 = imfilter(noisy_gaussian, h13);

filtered_sp_3 = imfilter(noisy_sp, h3);
filtered_sp_7 = imfilter(noisy_sp, h7);
filtered_sp_13 = imfilter(noisy_sp, h13);
```

```
figure;
subplot(4,4,1), imshow(original), title('Original');
subplot(4,4,2), imshow(filtered_gaussian_3), title('Gaussian 3x3');
subplot(4,4,3), imshow(filtered_gaussian_7), title('Gaussian 7x7');
subplot(4,4,4), imshow(filtered_gaussian_13), title('Gaussian 13x13');

subplot(4,4,5), imhist(original), title('Original Histogram');
subplot(4,4,6), imhist(filtered_gaussian_3), title('Gaussian 3x3 Histogram');
subplot(4,4,7), imhist(filtered_gaussian_7), title('Gaussian 7x7 Histogram');
subplot(4,4,8), imhist(filtered_gaussian_13), title('Gaussian 13x13 Histogram');

subplot(4,4,9), imshow(filtered_sp_3), title('Salt & Pepper 3x3');
subplot(4,4,10), imshow(filtered_sp_7), title('Salt & Pepper 7x7');
subplot(4,4,11), imshow(filtered_sp_13), title('Salt & Pepper 13x13');

subplot(4,4,13), imhist(filtered_sp_3), title('Salt & Pepper 3x3 Histogram');
subplot(4,4,14), imhist(filtered_sp_7), title('Salt & Pepper 7x7 Histogram');
subplot(4,4,15), imhist(filtered_sp_13), title('Salt & Pepper 13x13 Histogram');
```

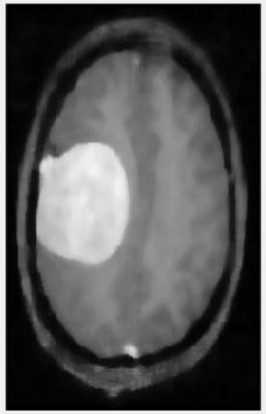
Original



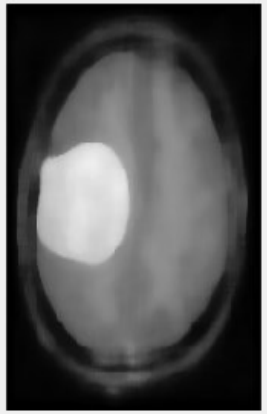
Gaussian Median 3x3



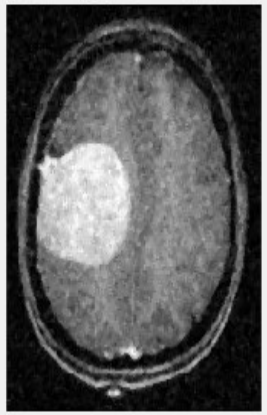
Gaussian Median 7x7



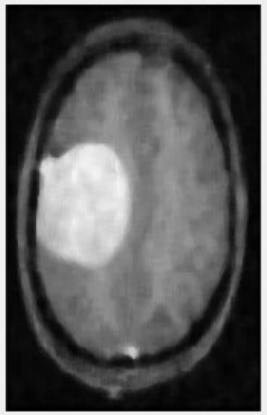
Gaussian Median 13x13



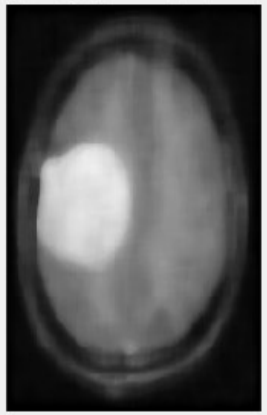
Salt & Pepper Median 3x3



Salt & Pepper Median 7x7



Salt & Pepper Median 13x13



```
% Appliquer le filtre médian sur les images bruitées
median_gaussian_3 = medfilt2(noisy_gaussian, [3 3]);
median_gaussian_7 = medfilt2(noisy_gaussian, [7 7]);
median_gaussian_13 = medfilt2(noisy_gaussian, [13 13]);

median_sp_3 = medfilt2(noisy_sp, [3 3]);
median_sp_7 = medfilt2(noisy_sp, [7 7]);
median_sp_13 = medfilt2(noisy_sp, [13 13]);
figure;
subplot(3,4,1), imshow(original), title('Original');
subplot(3,4,2), imshow(median_gaussian_3), title('Gaussian Median 3x3');
subplot(3,4,3), imshow(median_gaussian_7), title('Gaussian Median 7x7');
subplot(3,4,4), imshow(median_gaussian_13), title('Gaussian Median 13x13');

subplot(3,4,6), imshow(median_sp_3), title('Salt & Pepper Median 3x3');
subplot(3,4,7), imshow(median_sp_7), title('Salt & Pepper Median 7x7');
subplot(3,4,8), imshow(median_sp_13), title('Salt & Pepper Median 13x13');
```

Explication des effets des filtres

- **Filtre Moyenneur:** Il lisse l'image en remplaçant chaque pixel par la moyenne des pixels environnants, réduisant le bruit mais floutant les détails.
- **Filtre Médian:** Il remplace chaque pixel par la médiane des pixels environnants, ce qui est efficace pour supprimer le bruit impulsionnel comme le sel et poivre sans trop flouter l'image.