

IMN259 - Analyse d'images

Chapitre 3

Traitement d'images

Adapté des notes de cours de
Marie-Flavie Auclair-Fortier et Pierre-Marc Jodoin

OBJECTIFS DU THÈME

1. Se donner des outils de base qui servent dans diverses circonstances

- Ces outils sont des opérations qu'on effectue sur une image
- Ces outils seront réutilisés dans les chapitres suivants
- Certains de ces outils sont des rappels de IMN117 et IMN359...
- Pourquoi?
 - ✓ Augmenter la qualité visuelle
 - ✓ Améliorer les résultats d'une application autre
 - ◆ Prétraitement
- Types de dégradations
 - ✓ Origines matérielles
 - ◆ bruit, flou, support
 - ✓ Origine logicielle : introduite par un traitement
 - ◆ échantillonnage, quantification, compression ...
 - ✓ Origine sémantique : info superflue ayant un sens mais nuisant au traitement
 - ◆ mouvement, ombrage, éclairage, ...

PLAN

1. Opérations ponctuelles
2. Histogramme
3. Transformations géométriques
4. Bruit
 - 4.1. Rapport signal sur bruit
5. Opérations inter-images
 - 5.1. Vidéo et détection de mouvements

3. Transformations géométriques

Transformations géométriques

Jusqu'à présent, nous nous sommes concentré sur des procédés ayant pour but de modifier la distribution des niveaux de gris/couleurs de l'image. Dans cette sous-section, nous porterons notre attention sur des procédés ayant pour but de modifier la **distribution spatiale des pixels**.

Transformations géométriques

Transformation de base:



Translation



Cisaillement (*Shear*)



Rotation



Changement
d'échelle

Transformations géométriques

Transformations « artistiques » (le « *warping* »):



Ripples



Twirl



Zig Zag



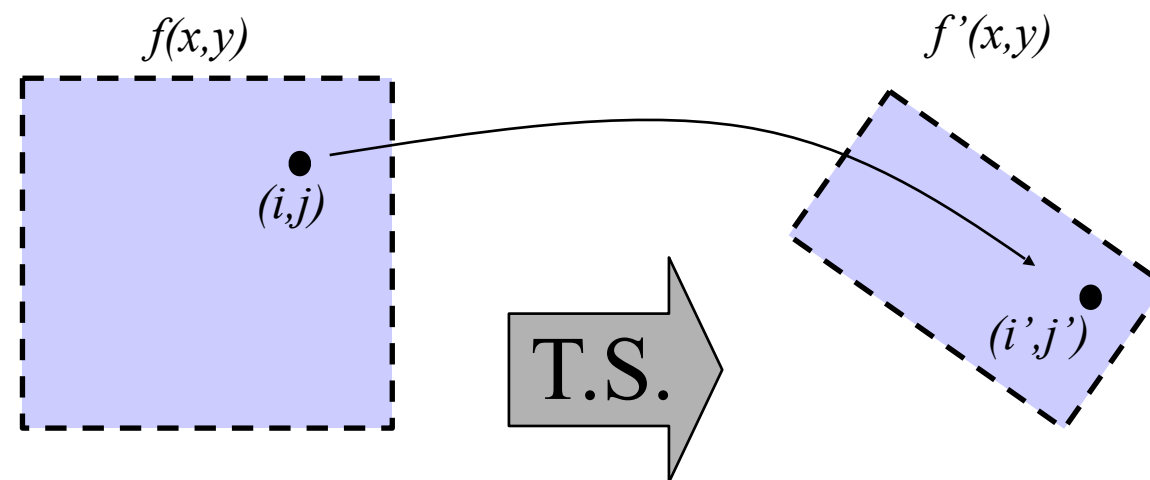
Lens



Transformations géométriques

Toute transformation géométrique est constituée de deux opérations de base:

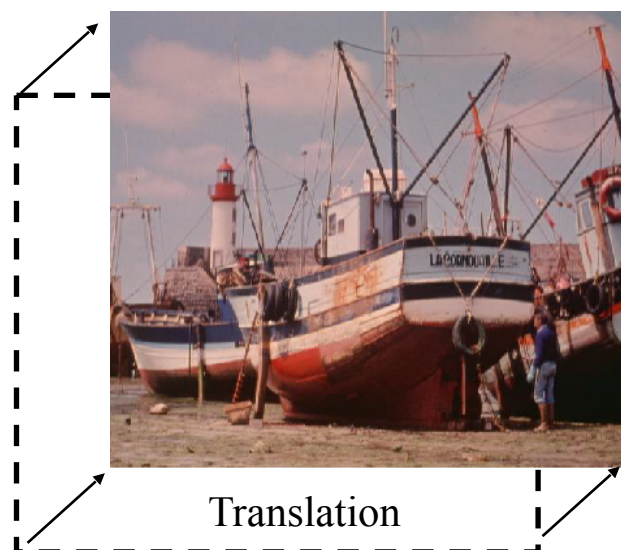
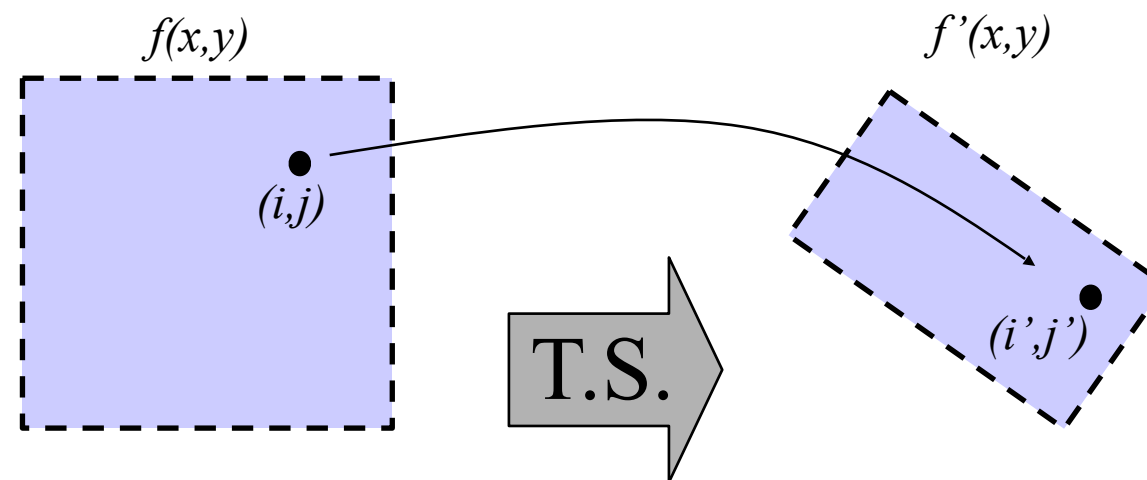
- 1- **transformation spatiale;**
- 2- interpolation de niveaux de gris.



Transformations géométriques

Toute transformation géométrique est constituée de deux opérations de base:

- 1- **transformation spatiale;**
- 2- interpolation de niveaux de gris.



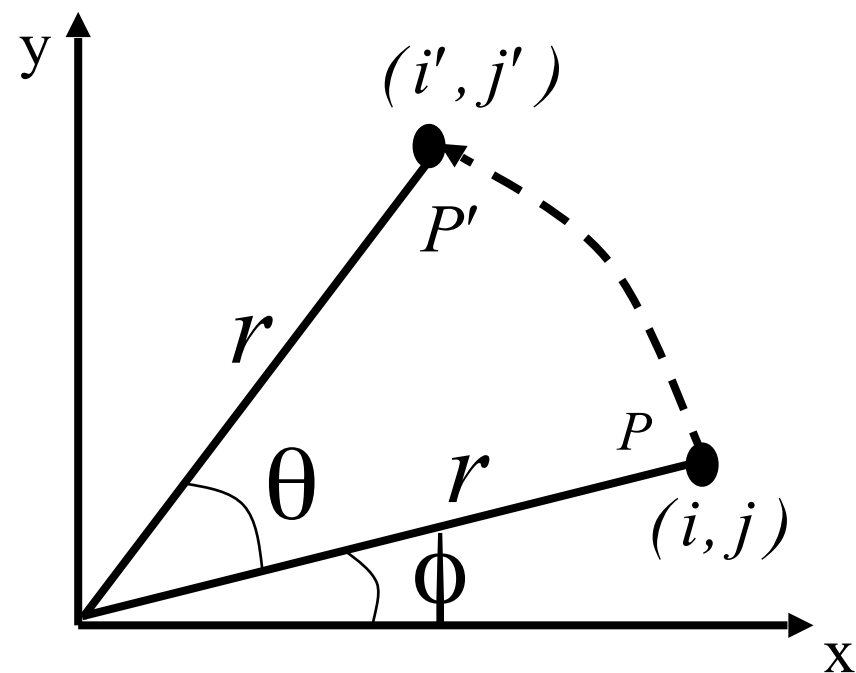
$$\begin{bmatrix} i' \\ j' \end{bmatrix} = \begin{bmatrix} \Delta x + i \\ \Delta y + j \end{bmatrix}$$

$$\begin{bmatrix} i' \\ j' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} i \\ j \end{bmatrix}$$

$$\begin{bmatrix} i' \\ j' \end{bmatrix} = \begin{bmatrix} \Delta x i \\ \Delta y j \end{bmatrix}$$

Transformations géométriques

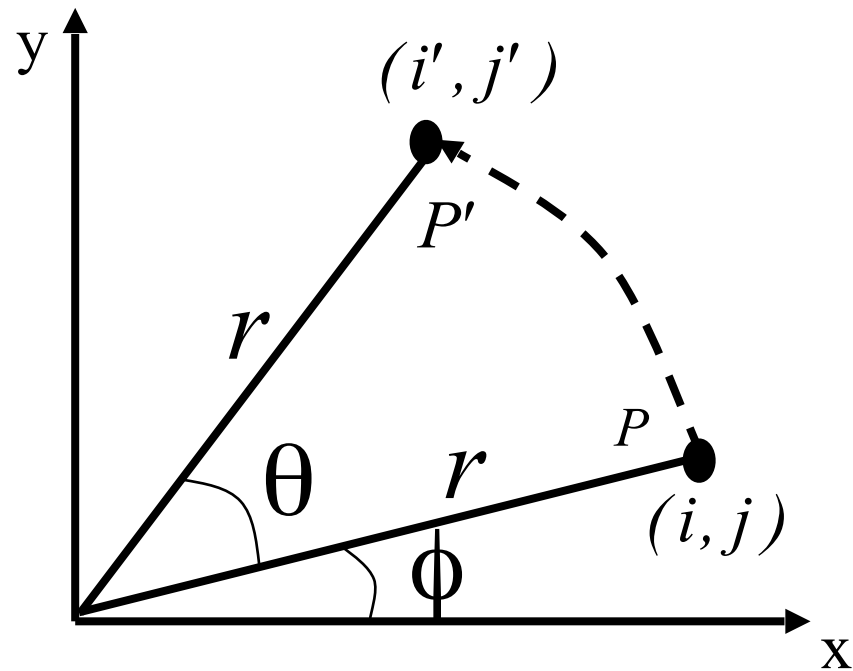
Rotation



θ positif = sens anti-horaire

Transformations géométriques

Rotation

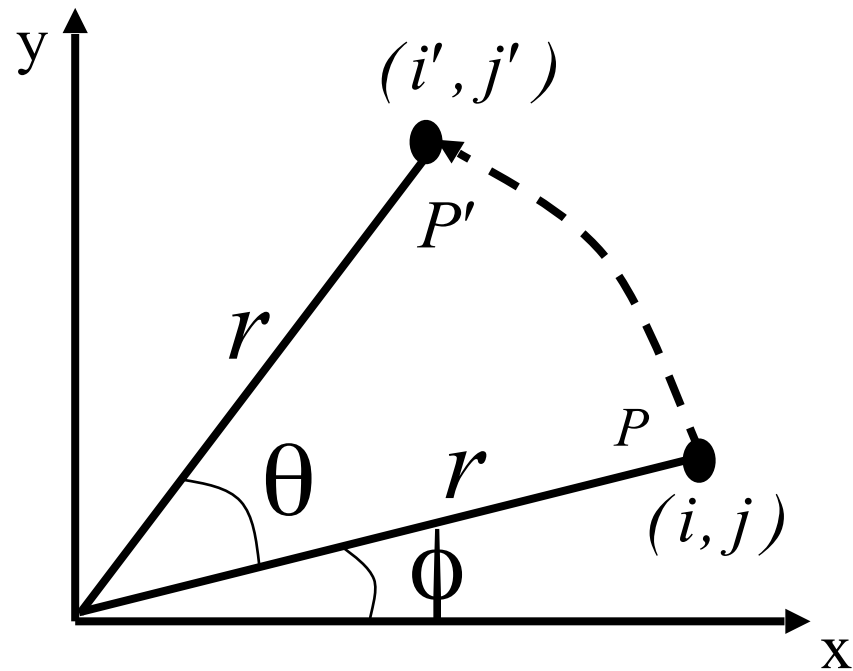


$$P : \begin{cases} i = r \cos \phi \\ j = r \sin \phi \end{cases}$$

θ positif = sens anti-horaire

Transformations géométriques

Rotation



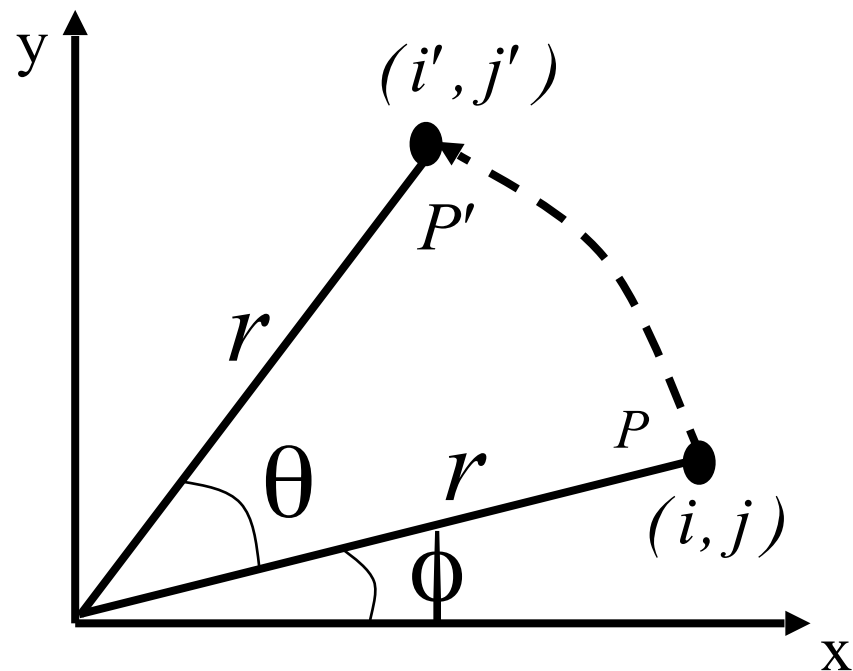
θ positif = sens anti-horaire

$$P : \begin{cases} i = r \cos \phi \\ j = r \sin \phi \end{cases}$$

$$P' : \begin{cases} i' = r \cos(\phi + \theta) \\ \quad = \underline{r \cos \phi} \cos \theta - \underline{r \sin \phi} \sin \theta \\ \quad = i \cos \theta - j \sin \theta \\ j' = r \sin(\phi + \theta) \\ \quad = \underline{r \cos \phi} \sin \theta + \underline{r \sin \phi} \cos \theta \\ \quad = i \sin \theta + j \cos \theta \end{cases}$$

Transformations géométriques

Rotation



θ positif = sens anti-horaire

$$\begin{bmatrix} i' \\ j' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} i \\ j \end{bmatrix}$$

$$P' = RP$$

$$P : \begin{cases} i = r \cos\phi \\ j = r \sin\phi \end{cases}$$

$$P' : \begin{cases} i' = r \cos(\phi + \theta) \\ \quad = \underline{r \cos\phi \cos\theta} - \underline{r \sin\phi \sin\theta} \\ \quad = i \cos\theta - j \sin\theta \\ j' = r \sin(\phi + \theta) \\ \quad = \underline{r \cos\phi \sin\theta} + \underline{r \sin\phi \cos\theta} \\ \quad = i \sin\theta + j \cos\theta \end{cases}$$

Transformations géométriques

Toute transformation géométrique est constituée de deux opérations de base:

- 1- **transformation spatiale;**
- 2- interpolation de niveaux de gris/couleur.

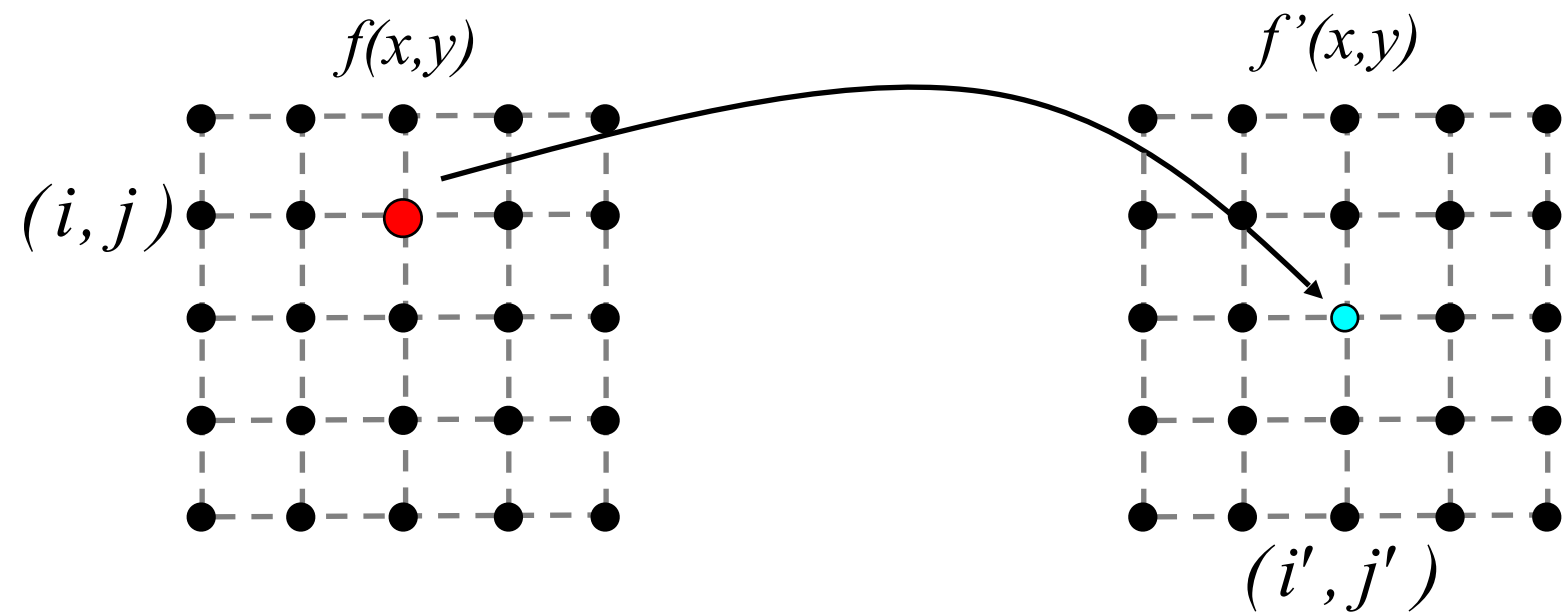


Wave

$$\begin{bmatrix} i' \\ j' \end{bmatrix} = \begin{bmatrix} i \\ j - 50 \sin\left(\frac{2\pi i}{100}\right) \end{bmatrix}$$

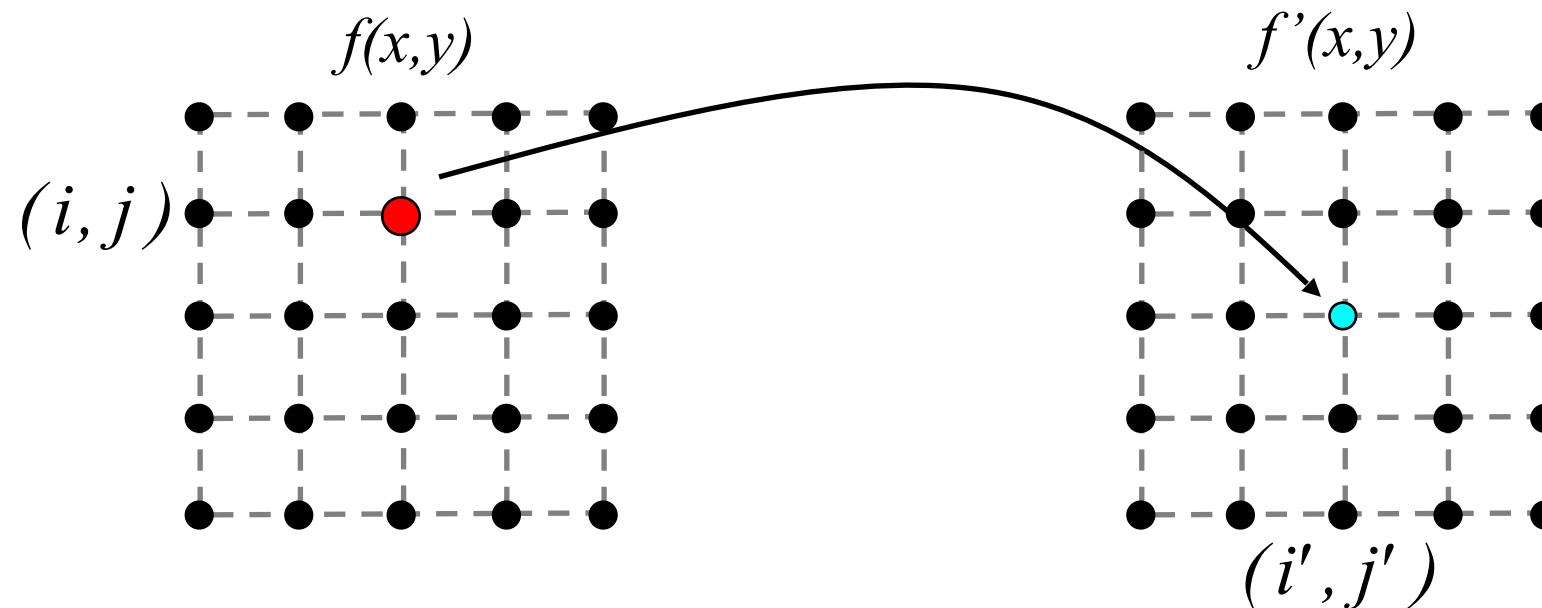
Transformations géométriques

Transformation spatiale directe



Transformations géométriques

Transformation spatiale **directe**



Contrairement à ce que l'intuition commande, la transformation spatiale directe est **À ÉVITER**. Pourquoi? Car elle peut laisser des trous béants dans l'image de sortie f' .

Exemple: changement d'échelle d'un facteur 5.

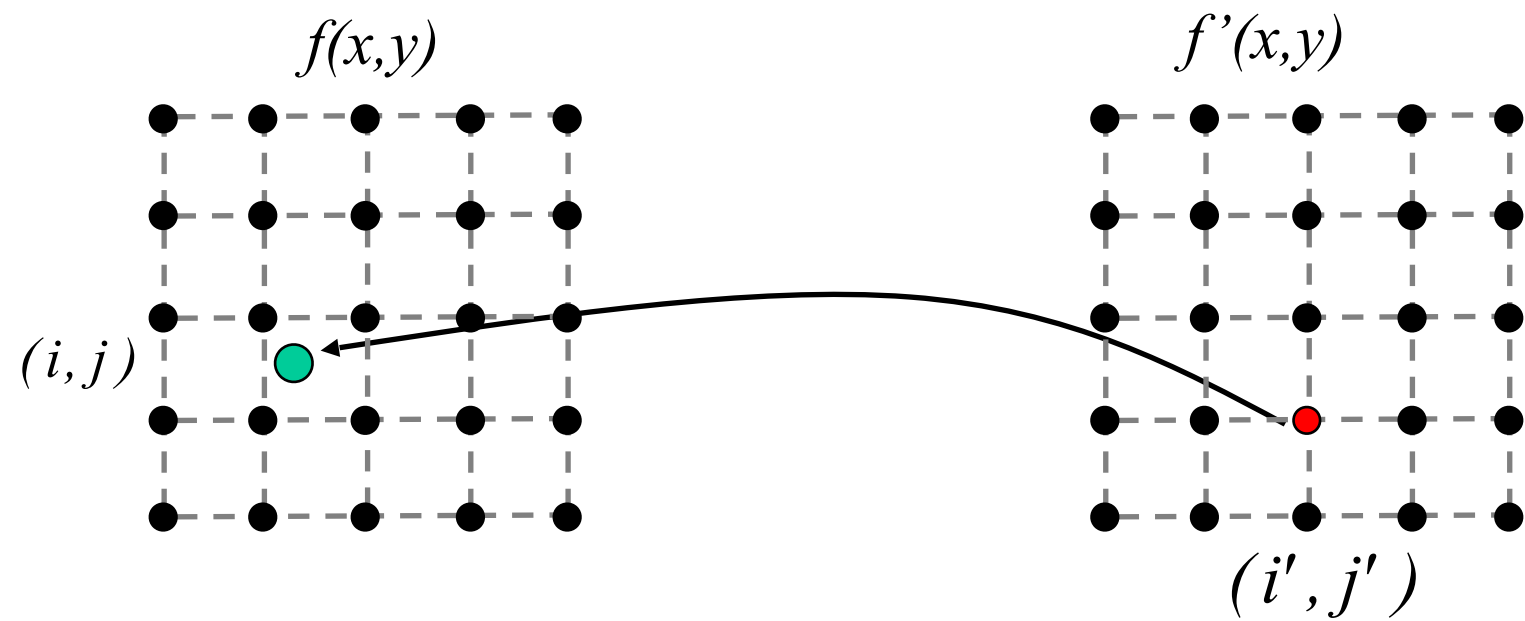
$$f(1,1) \rightarrow f'(5,1)$$

$$f(2,1) \rightarrow f'(10,1)$$

Et qu'y a-t-il entre $f'(5,1)$ et $f'(10,1)$? En mode **directe**, RIEN!!!!

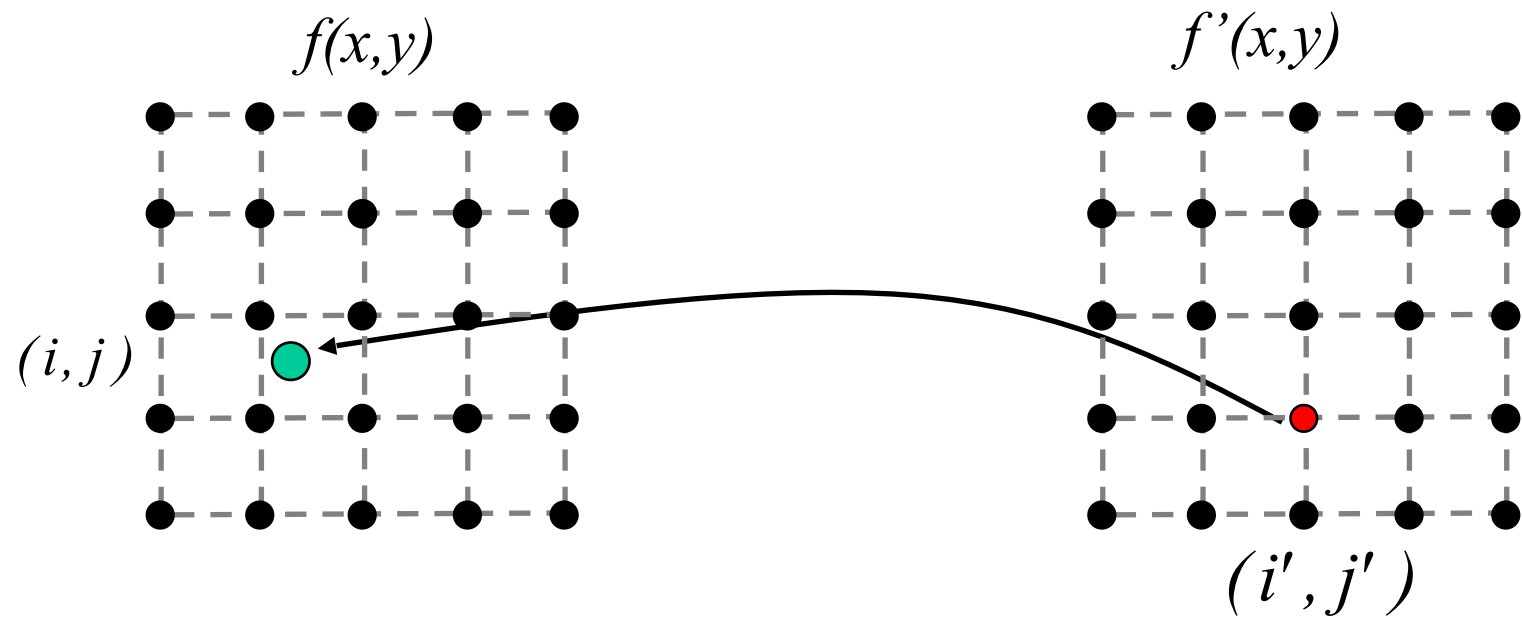
Transformations géométriques

Transformation spatiale inverse



Transformations géométriques

Transformation spatiale inverse

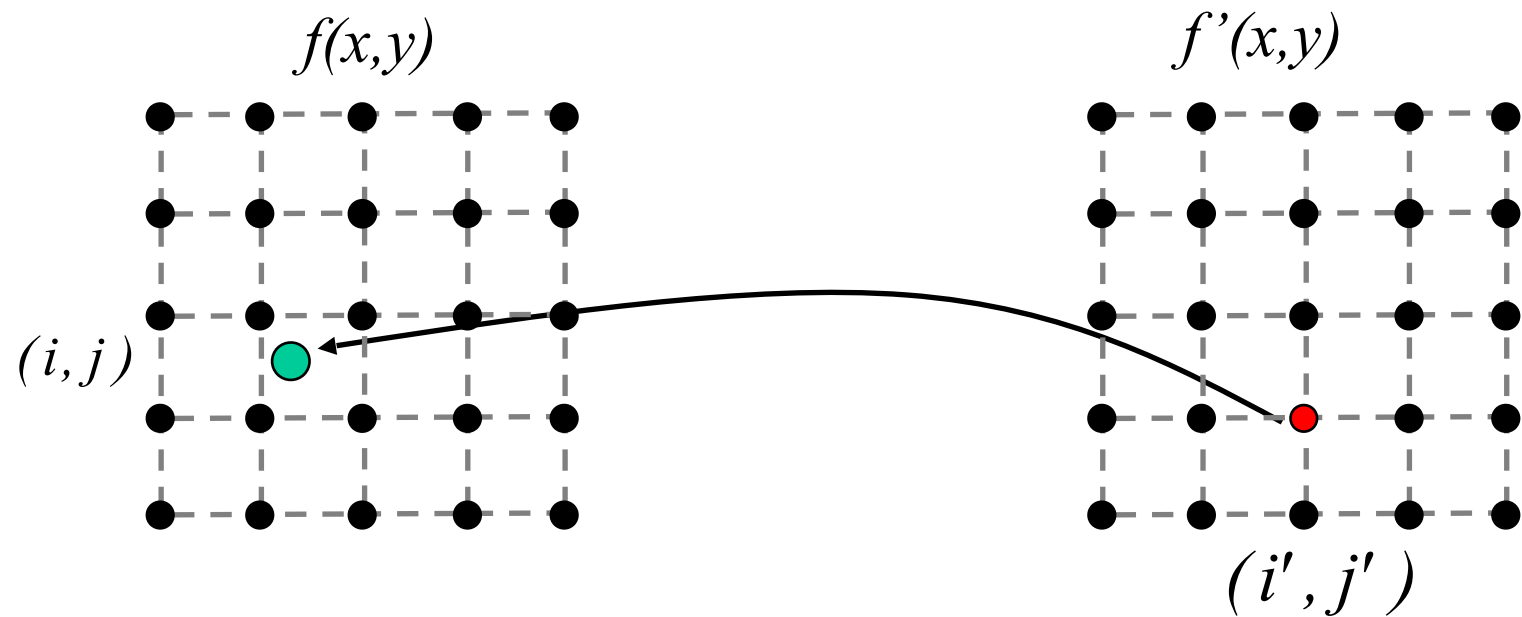


De cette façon, on pose la question suivante :

« Quelle est la couleur du pixel (i',j') ? Réponse: c'est $f(i,j)$! »

Transformations géométriques

Transformation spatiale inverse



De cette façon, on pose la question suivante :

« Quelle est la couleur du pixel (i', j') ? Réponse: c'est $f(i, j)$! »

Grâce à cette méthode, on évite les trous. Toutefois, le pixel (i, j) possède rarement des coordonnées entières. Par exemple, que faire lorsque (i', j') pointe vers

$$f(20.2, 44.9)?$$

Je connais $f(20, 44)$, $f(21, 44)$, $f(20, 45)$ et $f(21, 45)$ mais pas $f(20.2, 44.9)$.

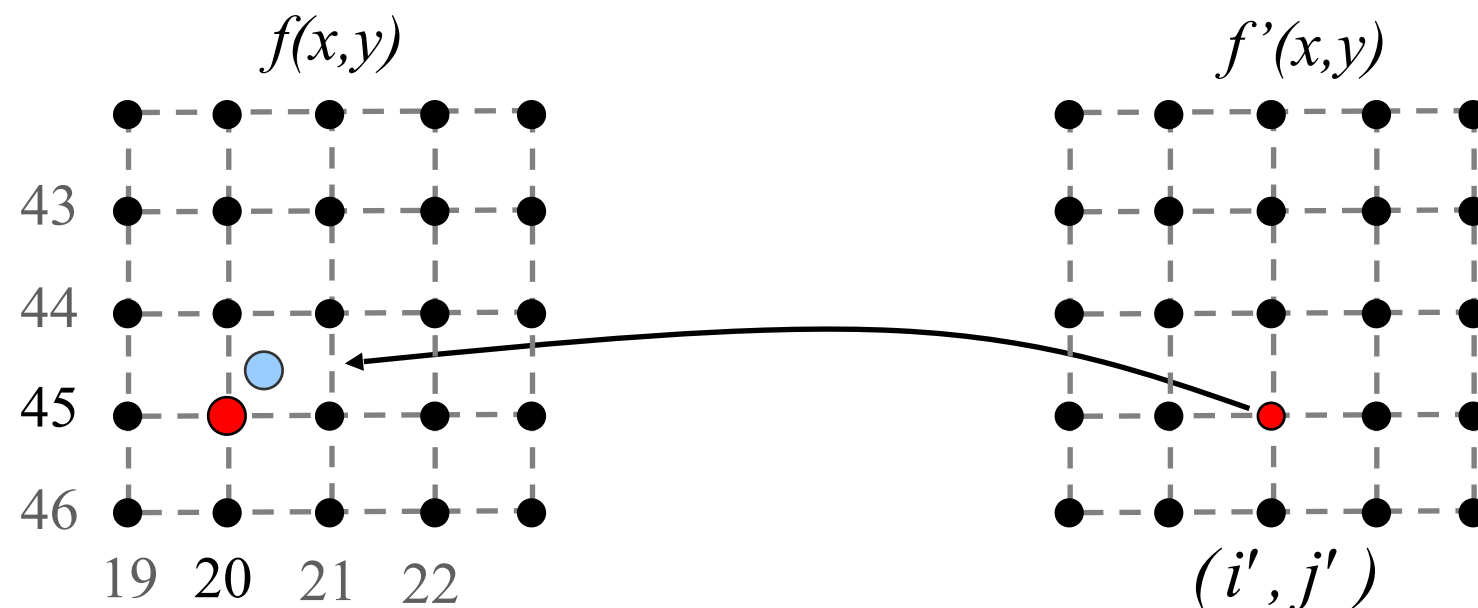
Transformations géométriques

Transformation spatiale inverse

- 1- transformation spatiale;
- 2- **interpolation de niveaux de gris.**

Parmi les [très] nombreuses méthodes d'interpolation, trois sont fréquemment sollicitées en imagerie.

A. Le pixel le plus près



$$(i, j) = (20.2, 44.9) \rightarrow (20, 45)$$

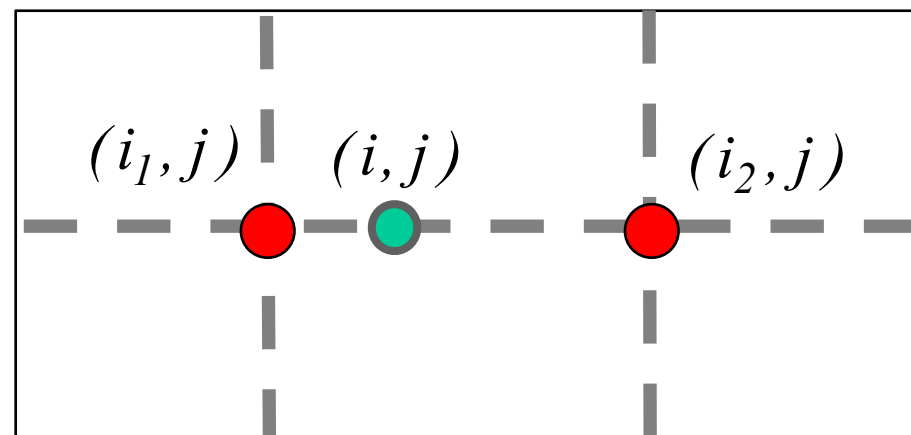
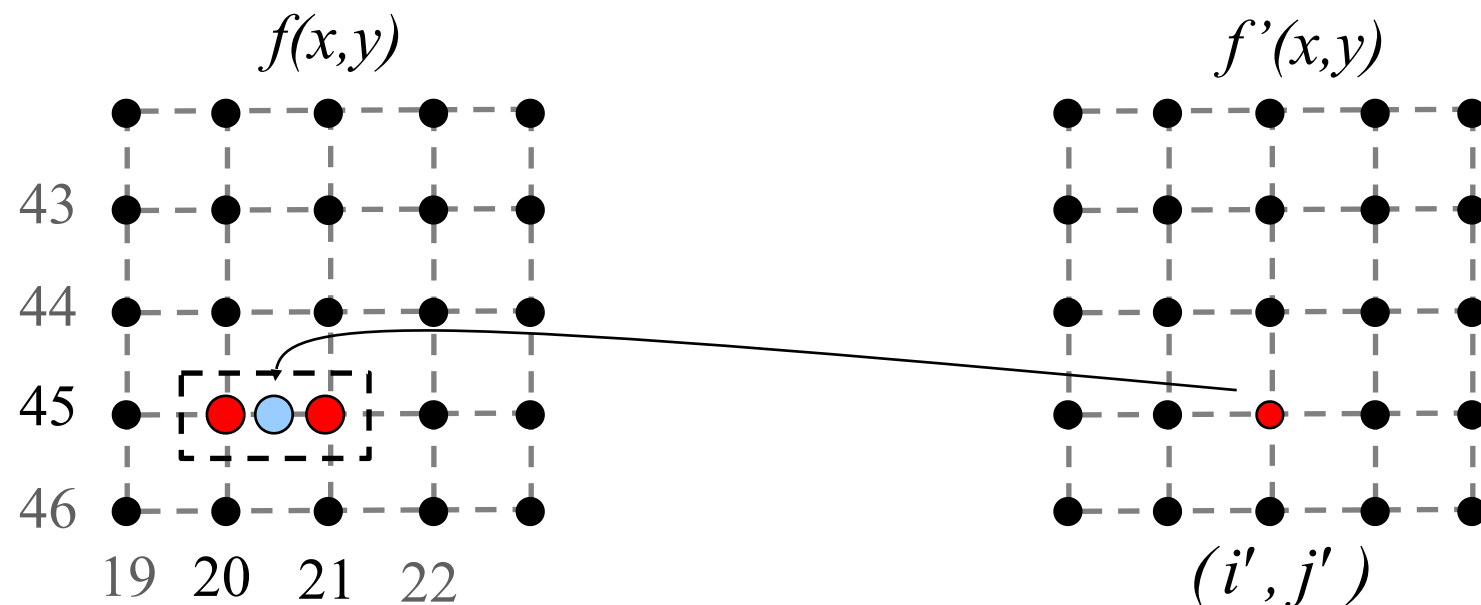
Solution simple mais qui induit des imprécisions.

Transformations géométriques

Transformation spatiale inverse

- 1- transformation spatiale;
- 2- **interpolation de niveaux de gris.**

B. Interpolation linéaire (si (i,j) tombe entre deux pixels)



$$f(i, j) \approx f(i_1, j)(i_2 - i) + f(i_2, j)(i - i_1)$$

$$\text{si } (i, j) = (20.2, 45)$$

$$f(20.2, 45) \approx f(20, 45) \times 0.8 + f(21, 45) \times 0.2$$

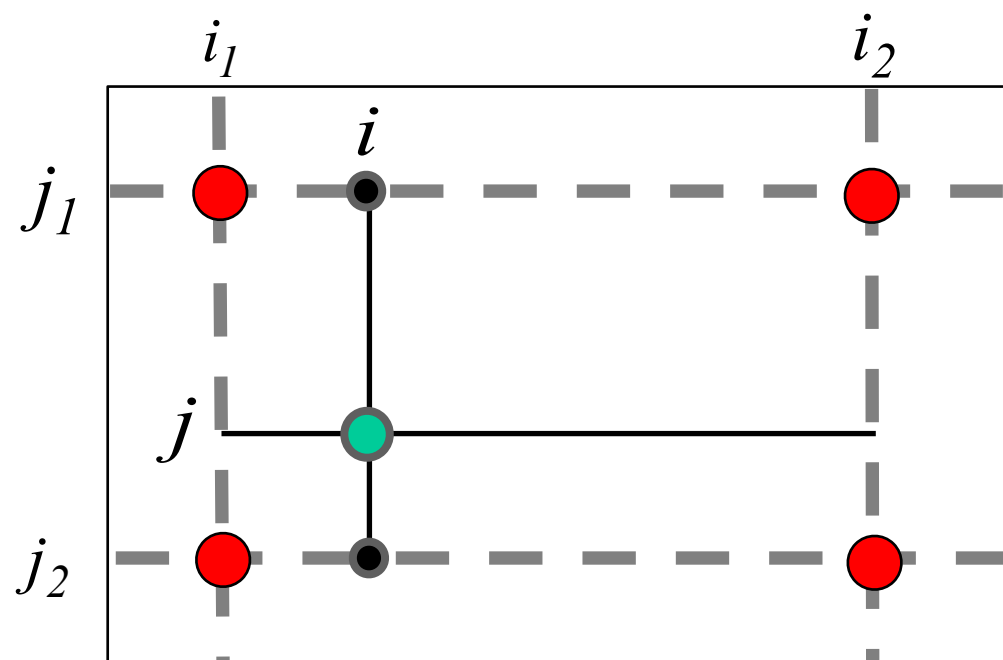
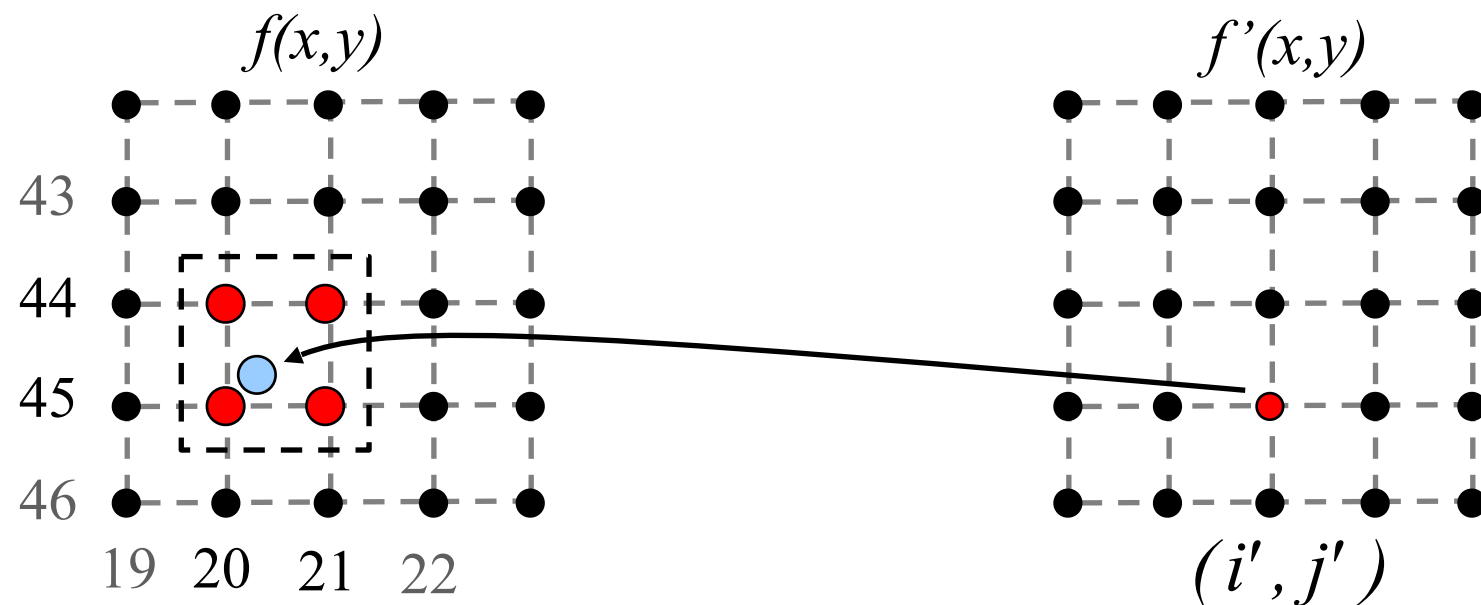
Transformations géométriques

Transformation spatiale inverse

1- transformation spatiale;

2- **interpolation de niveaux de gris.**

C. **Interpolation bilinéaire** (si (i,j) tombe entre quatre pixels)



$$f(i, j_1) \approx f(i_1, j_1)(i_2 - i) + f(i_2, j_1)(i - i_1)$$

$$f(i, j_2) \approx f(i_1, j_2)(i_2 - i) + f(i_2, j_2)(i - i_1)$$

$$f(i, j) \approx f(i, j_1)(j_2 - j) + f(i, j_2)(j - j_1)$$

Transformations géométriques

Transformation spatiale inverse

1- transformation spatiale;

2- **interpolation de niveaux de gris.**

$$f'(x,y)$$

$$f(x,y)$$



Plus proche voisin



Interpolation bilinéaire



Une autre transformation géométrique :
le changement d'échelle

Transformations géométriques

Changement d'échelle
(*image resizing*)

$$\begin{bmatrix} M' \\ N' \end{bmatrix} = \begin{bmatrix} \Delta x M \\ \Delta y N \end{bmatrix}$$

où M, N est la dimension de l'image.

Dans le cas d'un changement d'échelle, le pixel destination (i, j) ne possède pas la même taille que le pixel d'origine (i', j') .

Lorsque $\Delta x, \Delta y \geq 1$ une interpolation bilinéaire marche bien.

$$\begin{array}{|c|c|c|c|} \hline & f(x,y) & & f'(x,y) \\ \hline \end{array}$$

(i', j')	(i, j)
$(0, 0)$	$(0, 0)$
$(1, 0)$	$(0.5, 0)$
$(2, 0)$	$(1, 0)$
$(1, 1)$	$(0.5, 0.5)$
...	...

Transformations géométriques

Changement d'échelle
(*image resizing*)

$$\begin{bmatrix} M \\ N \end{bmatrix} = \begin{bmatrix} 0.5M' \\ 0.5N' \end{bmatrix}$$

où M, N est la dimension de l'image.

	$f(x,y)$		
	0	1	2
0	a	b	c
1	d	e	f
2	g	h	i

	$f'(x,y)$					
	0	1	2	3	4	5
0	a	$\frac{a+b}{2}$	b			
1	$\frac{a+d}{2}$	A	$\frac{b+e}{2}$			
2	d	$\frac{d+e}{2}$	e			
3						
4						
5						

$$A = \frac{a + b + d + e}{4}$$

Transformations géométriques

Changement d'échelle
(*image resizing*) $\begin{bmatrix} M' \\ N' \end{bmatrix} = \begin{bmatrix} \Delta x M \\ \Delta y N \end{bmatrix}$ où M, N est la dimension de l'image.

Dans le cas d'un changement de résolution, le pixel destination (i, j) ne possède pas la même taille que le pixel d'origine (i', j') .

Par contre, lorsque $0 < \Delta x, \Delta y < 1$, il nous faut trouver une autre méthode.

$$\begin{array}{c} f(x, y) \\ \begin{array}{|c|c|c|c|c|c|c|c|} \hline \square & \square & \square & \square & \square & \square & \square & \square \\ \hline \square & \square & \square & \square & \square & \square & \square & \square \\ \hline \square & \square & \square & \square & \square & \square & \square & \square \\ \hline \square & \square & \square & \square & \square & \square & \square & \square \\ \hline \square & \square & \square & \square & \square & \square & \square & \square \\ \hline \square & \square & \square & \square & \square & \square & \square & \square \\ \hline \square & \square & \square & \square & \square & \square & \square & \square \\ \hline \end{array} \end{array} \times \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} = \begin{array}{c} f'(x, y) \\ \begin{array}{|c|c|c|c|} \hline \square & \square & \square & \square \\ \hline \square & \square & \square & \square \\ \hline \square & \square & \square & \square \\ \hline \square & \square & \square & \square \\ \hline \end{array} \end{array}$$

Pourquoi une autre méthode? Voici un exemple.

Transformations géométriques

Changement d'échelle
(*image resizing*)

$$\begin{bmatrix} M \\ N \end{bmatrix} = \begin{bmatrix} 2M' \\ 2N' \end{bmatrix}$$

où M, N est la dimension de l'image.

La façon la plus simple est de prendre le voisin le plus proche.

(i', j')	(i, j)
$(0, 0)$	$(0, 0)$
$(1, 0)$	$(2, 0)$
$(2, 0)$	$(4, 0)$
...	...

	0	1	2	3	4	5
0	a	b	c	d	e	f
1	g	g	i	j	k	l
2	m	n	o	p	q	r
3	s	t	u	v	w	x
4	y	z	α	β	χ	δ
5	ε	ϕ	γ	η	ι	φ

	0	1	2
0	a	c	e
1	m	o	q
2	y	α	χ

Transformations géométriques

Changement d'échelle
(*image resizing*)

$$\begin{bmatrix} M' \\ N' \end{bmatrix} = \begin{bmatrix} 0.5M \\ 0.5N \end{bmatrix}$$

où M, N est la dimension de l'image.

$$\begin{bmatrix} M \\ N \end{bmatrix} = \begin{bmatrix} 2M' \\ 2N' \end{bmatrix}$$

La façon la plus simple est de prendre le voisin le plus proche.

(i', j')	(i, j)
$(0, 0)$	$(0, 0)$
$(1, 0)$	$(2, 0)$
$(2, 0)$	$(4, 0)$
...	...

Transformations géométriques

Changement d'échelle
(*image resizing*)

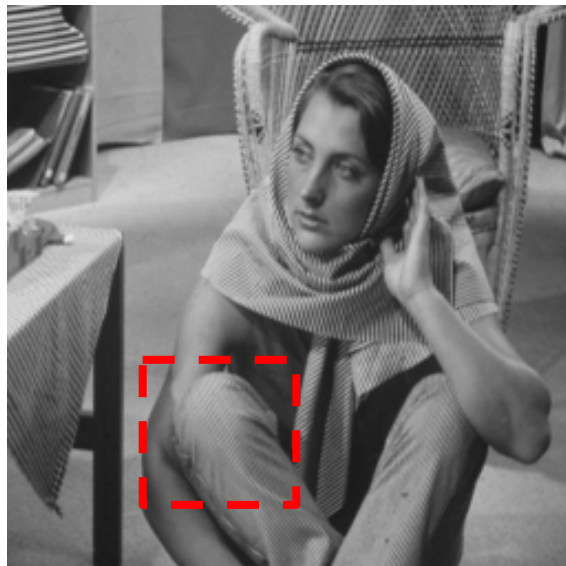
$$\begin{bmatrix} M' \\ N' \end{bmatrix} = \begin{bmatrix} 0.5M \\ 0.5N \end{bmatrix}$$

où M, N est la dimension de l'image.

$$\begin{bmatrix} M \\ N \end{bmatrix} = \begin{bmatrix} 2M' \\ 2N' \end{bmatrix}$$

La façon la plus simple est de prendre le voisin le plus proche.

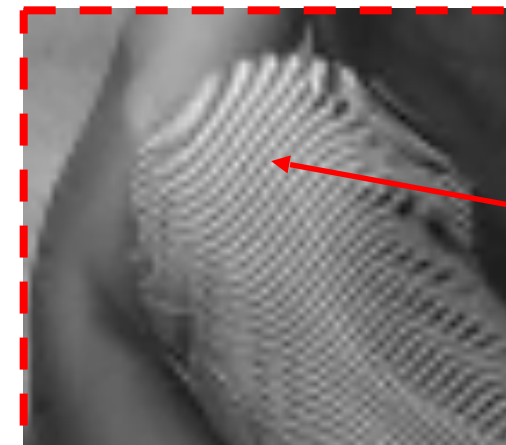
(i', j')	(i, j)
$(0, 0)$	$(0, 0)$
$(1, 0)$	$(2, 0)$
$(2, 0)$	$(4, 0)$
...	...



$f(x, y)$



$f'(x, y)$



Aliassing
(Moiré)

Transformations géométriques

Changement d'échelle
(*image resizing*)

$$\Delta x, \Delta y = 0.5$$

$$\begin{bmatrix} M' \\ N' \end{bmatrix} = \begin{bmatrix} 0.5M \\ 0.5N \end{bmatrix}$$

où M, N est la dimension de l'image.

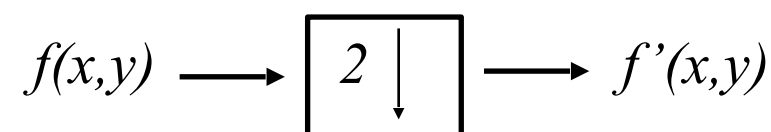
$$\begin{bmatrix} M \\ N \end{bmatrix} = \begin{bmatrix} 2M' \\ 2N' \end{bmatrix}$$

La façon la plus simple est de prendre le voisin le plus proche.

(i', j')	(i, j)
$(0, 0)$	$(0, 0)$
$(1, 0)$	$(2, 0)$
$(2, 0)$	$(4, 0)$
...	...

Les pixels aux coordonnées impaires dans $f(x, y)$ ne sont pas sollicités. Une telle approche porte le nom de « **décimation** » lorsque

$$\Delta x = \frac{1}{P} \quad \text{et} \quad \Delta y = \frac{1}{R} \quad \text{où } P, R \in \mathbb{N}^+$$



Transformations géométriques

Changement d'échelle
(*image resizing*)

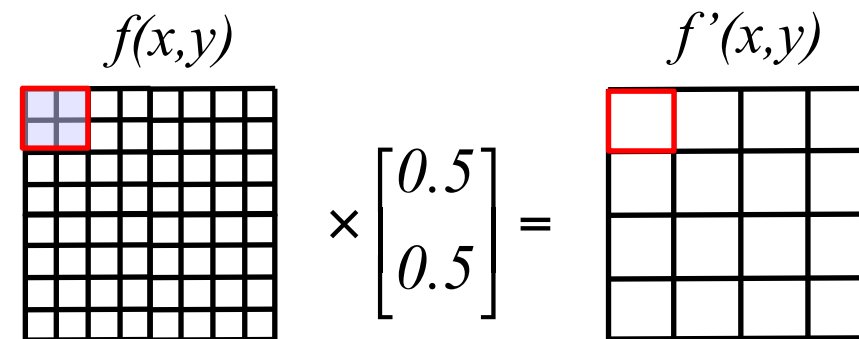
$$\Delta x, \Delta y = 0.5$$

$$\begin{bmatrix} M' \\ N' \end{bmatrix} = \begin{bmatrix} 0.5M \\ 0.5N \end{bmatrix}$$

où M, N est la dimension de l'image.

$$\begin{bmatrix} M \\ N \end{bmatrix} = \begin{bmatrix} 2M' \\ 2N' \end{bmatrix}$$

En fait, lorsque $\Delta x, \Delta y = 0.5$ un pixel (i', j') couvre quatre pixels dans $f(x, y)$



Par conséquent, lorsque $0 < \Delta x, \Delta y \leq 1$

$$f'(x, y) = \text{Moyenne des pixels couverts par } (x, y) \text{ dans } f(x, y)$$

Transformations géométriques

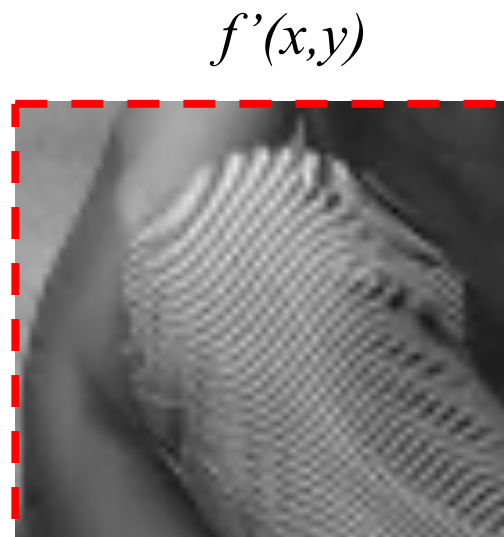
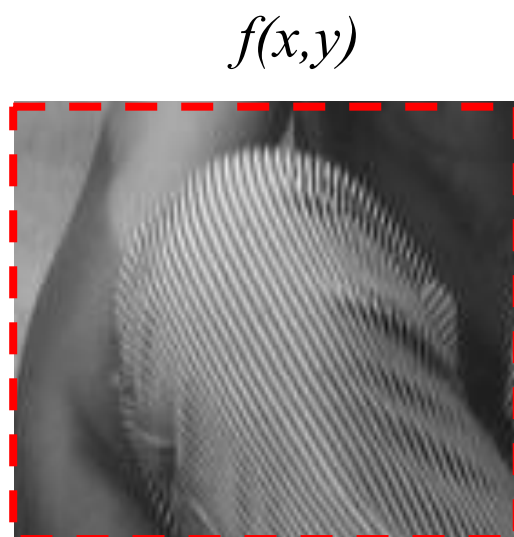
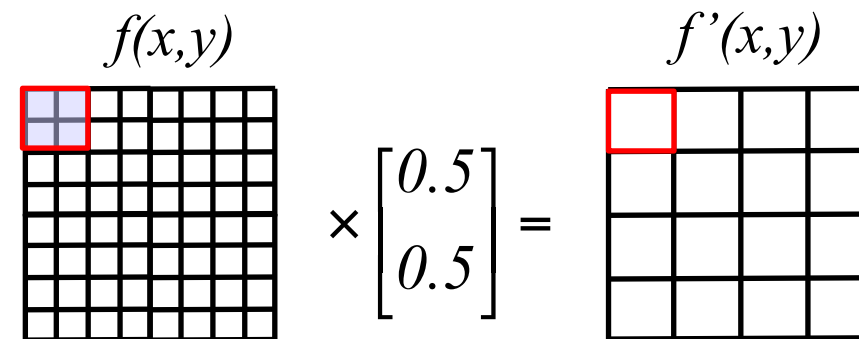
Changement d'échelle
(*image resizing*)

$$\begin{bmatrix} M' \\ N' \end{bmatrix} = \begin{bmatrix} 0.5M \\ 0.5N \end{bmatrix}$$

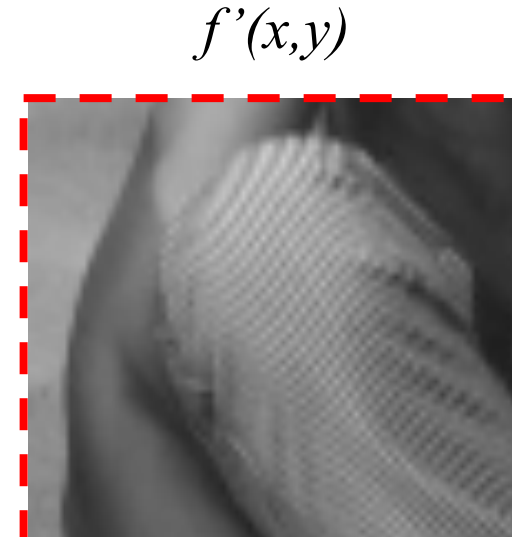
où M, N est la dimension de l'image.

$$\begin{bmatrix} M \\ N \end{bmatrix} = \begin{bmatrix} 2M' \\ 2N' \end{bmatrix}$$

En fait, lorsque $\Delta x, \Delta y = 0.5$ un pixel (i', j') couvre quatre pixels dans $f(x, y)$



plus proche voisin



moyenne

Transformations géométriques

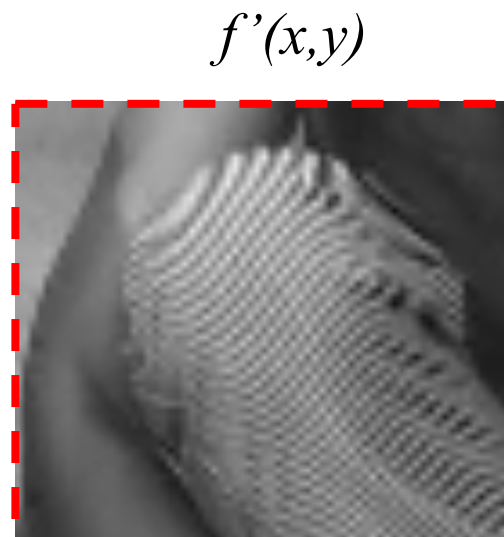
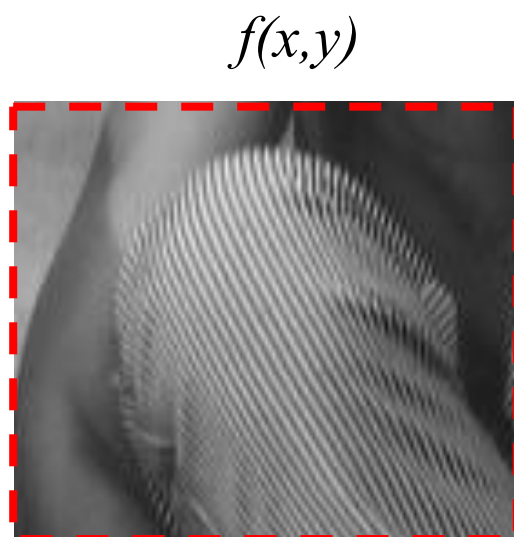
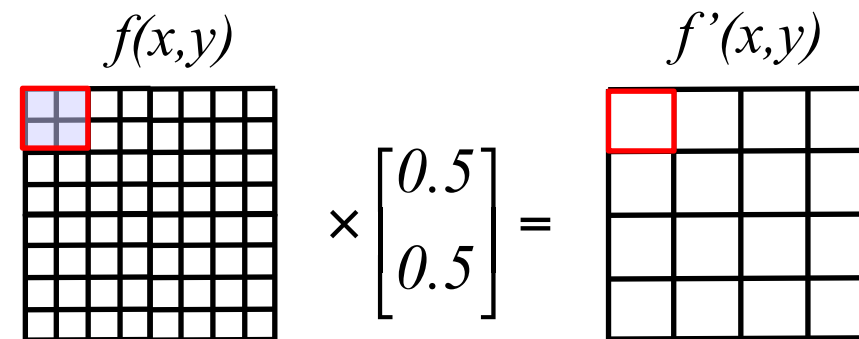
Changement d'échelle
(*image resizing*)

$$\begin{bmatrix} M' \\ N' \end{bmatrix} = \begin{bmatrix} 0.5M \\ 0.5N \end{bmatrix}$$

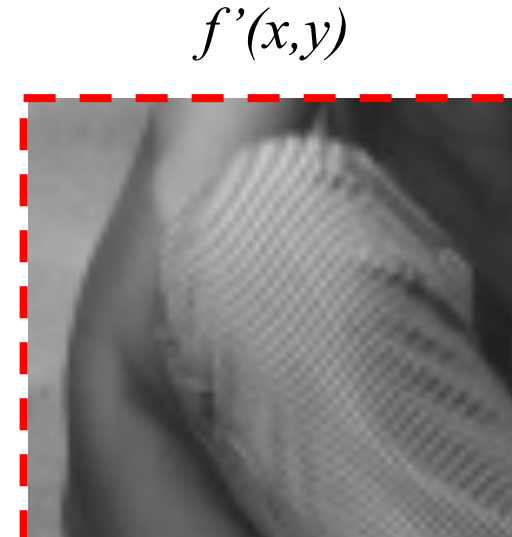
où M, N est la dimension de l'image.

$$\begin{bmatrix} M \\ N \end{bmatrix} = \begin{bmatrix} 2M' \\ 2N' \end{bmatrix}$$

En fait, lorsque $\Delta x, \Delta y = 0.5$ un pixel (i', j') couvre quatre pixels dans $f(x, y)$



plus proche voisin



moyenne

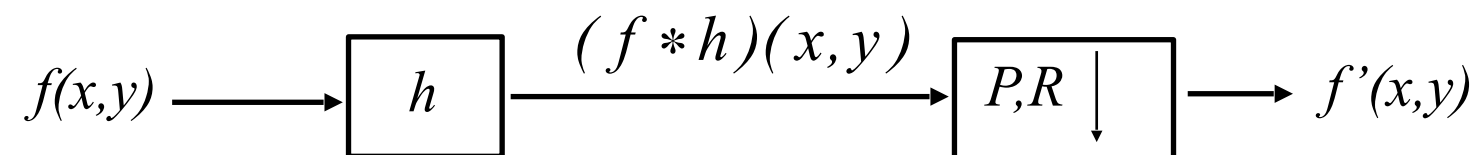
Outils mathématiques
et d'analyses avancés
pour améliorer le
résultat!

(IMN764)

Transformations géométriques

Changement d'échelle

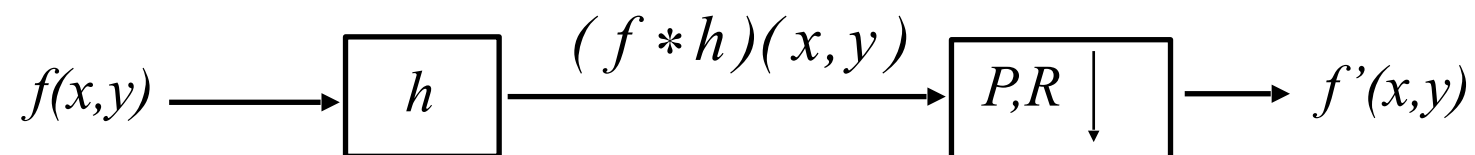
Lorsque $\Delta x = \frac{l}{P}$ et $\Delta y = \frac{l}{R}$ où $P, R \in \mathbb{N}^+$ une façon simple de calculer « la moyenne des pixels couverts par (x,y) » est de filtrer $f(x,y)$ par un filtre passe-bas et de décimer l'image filtrée.



Transformations géométriques

Changement d'échelle

Lorsque $\Delta x = \frac{l}{P}$ et $\Delta y = \frac{l}{R}$ où $P, R \in \mathbb{N}^+$ une façon simple de calculer « la moyenne des pixels couverts par (x, y) » est de filtrer $f(x, y)$ par un filtre passe-bas et de décimer l'image filtrée.



Les deux filtres les plus couramment utilisés sont les filtres:

- Moyeneur de taille : $\frac{l}{\Delta x} \times \frac{l}{\Delta y}$
- Gaussien d'écart type est : $\sigma_x = \frac{l}{4\Delta x}$, $\sigma_y = \frac{l}{4\Delta y}$

Transformations géométriques

Changement de résolution

Lorsque $\Delta x = \frac{1}{P}$ et $\Delta y = \frac{1}{R}$ où $P, R \in \mathbb{N}^+$ une façon simple de calculer « la moyenne

des pixels couverts par (x,y) » est de filtrer $f(x,y)$ par un filtre passe-bas et de décimer l'image filtrée.

$f(x,y)$



$\Delta x = \Delta y = 0.5$

plus proche voisin



$f'(x,y)$
Filtre moyenneur



Filtre gaussien



$\Delta x = \Delta y = 0.25$



Transformations géométriques

Changement de résolution

Lorsque $\Delta x = \frac{1}{P}$ et $\Delta y = \frac{1}{R}$ où $P, R \in \mathbb{N}^+$ une façon simple de calculer « la moyenne

des pixels couverts par (x, y) » est de filtrer $f(x, y)$ par un filtre passe-bas et de décimer l'image filtrée.

$f(x, y)$



$\Delta x = \Delta y = 0.5$

plus proche voisin



$f'(x, y)$
Filtre moyennneur



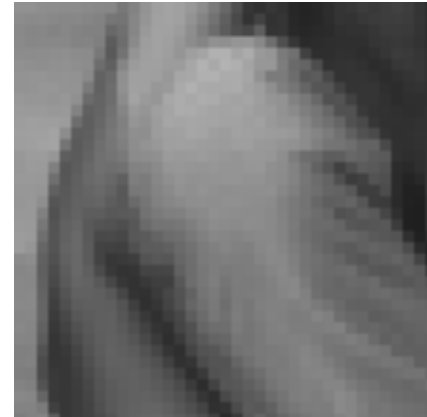
Filtre gaussien



Outils mathématiques
et d'analyses avancés
pour améliorer le
résultat!

$\Delta x = \Delta y = 0.25$

(IMN764)



PLAN

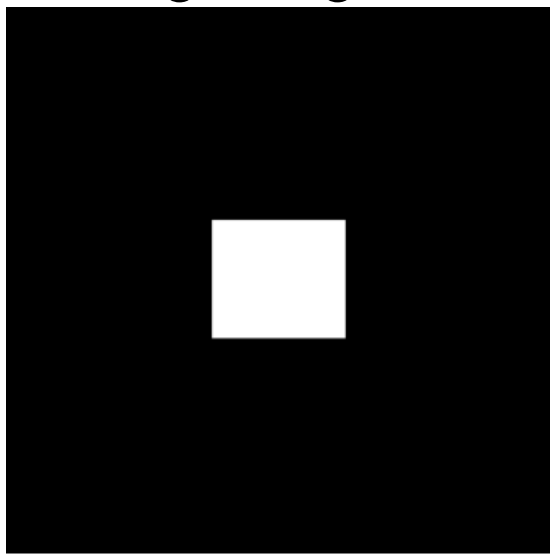
1. Opérations ponctuelles
2. Histogramme
3. Transformations géométriques
4. Bruit
 - 4.1. Rapport signal sur bruit
5. Opérations inter-images
 - 5.1. Vidéo et détection de mouvements

Le bruit dans une image, c'est quoi?

Demo 03

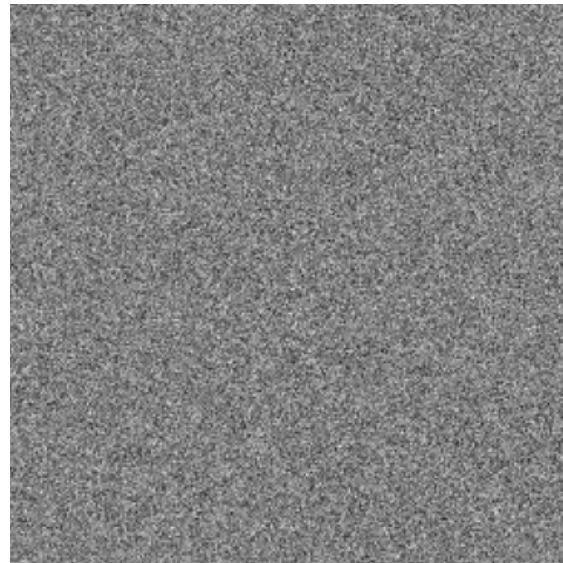
*Ici, bruit Gaussien
(moyenne 0 et écart type 0.2)*

Images originales

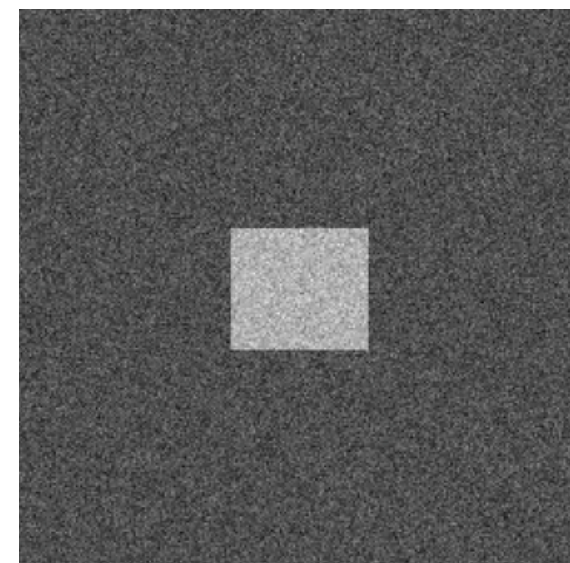


$f(x,y)$

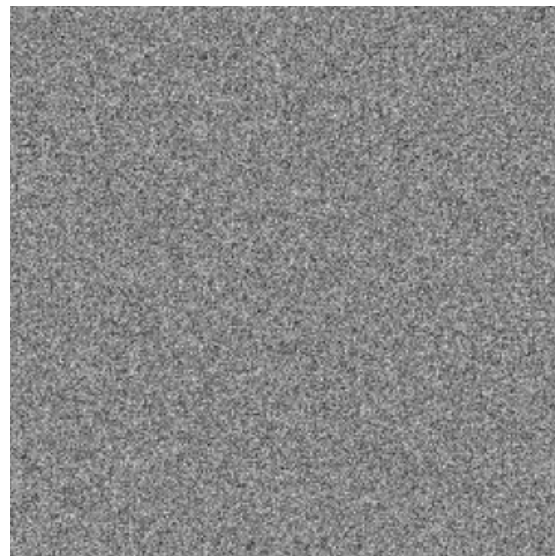
+



$\eta(x,y)$



$= g(x,y)$



*`np.random.normal`
`(0, 0.2, img.shape)`*



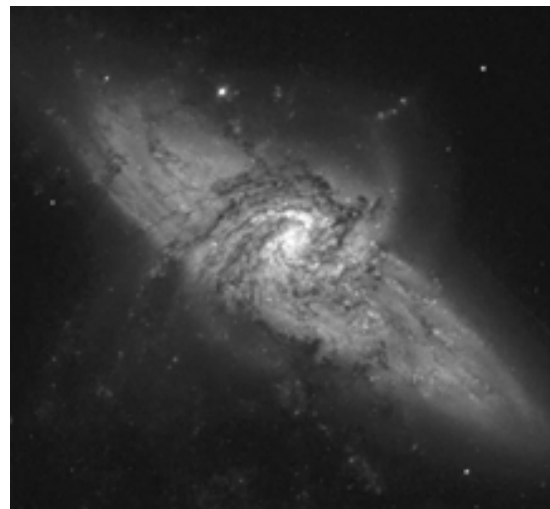
$$g(x,y) = f(x,y) + \eta(x,y)$$

$g(x,y)$ a été obtenue en additionnant un bruit $\eta(x,y)$ **non corrélé** et de **moyenne nulle**.

Opérations inter-images.

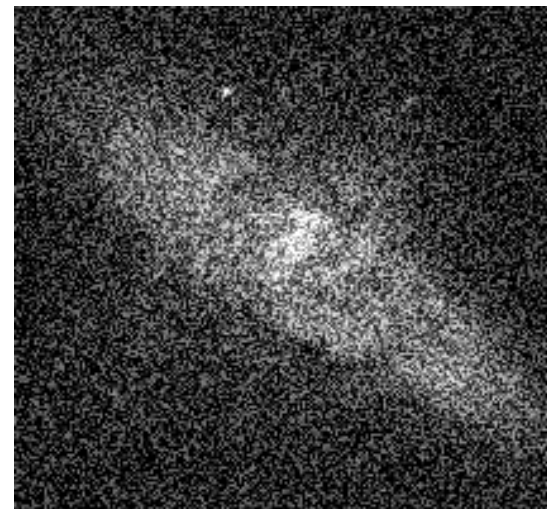
Réduction du bruit par moyennage

Image originale



$f(x,y)$

Image corrompue par
du bruit numérique



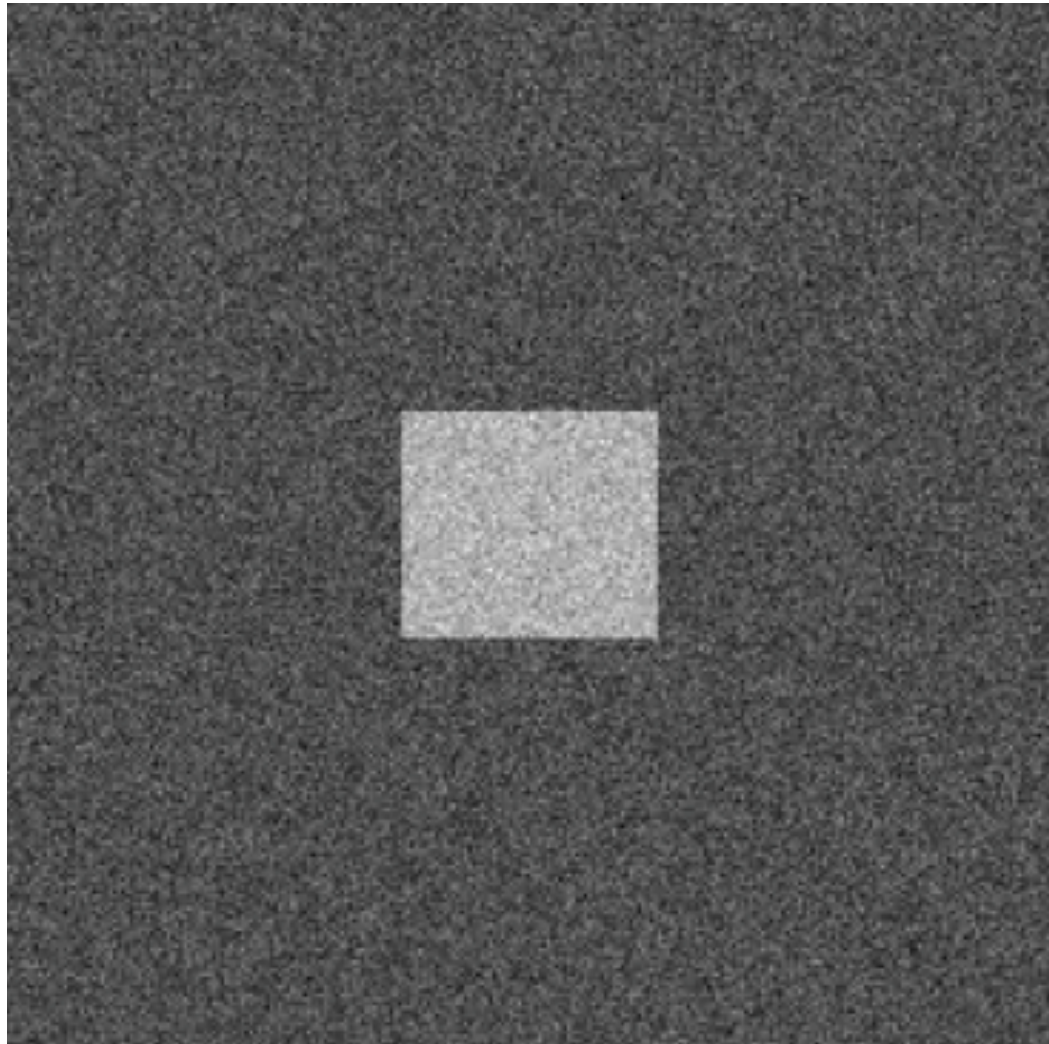
$g(x,y)$

$$g(x,y) = f(x,y) + \eta(x,y)$$

$g(x,y)$ a été obtenue en additionnant un bruit $\eta(x,y)$ **non corrélé** et de **moyenne nulle**.

Rapport signal sur bruit (SNR) en pratique

Demo 03

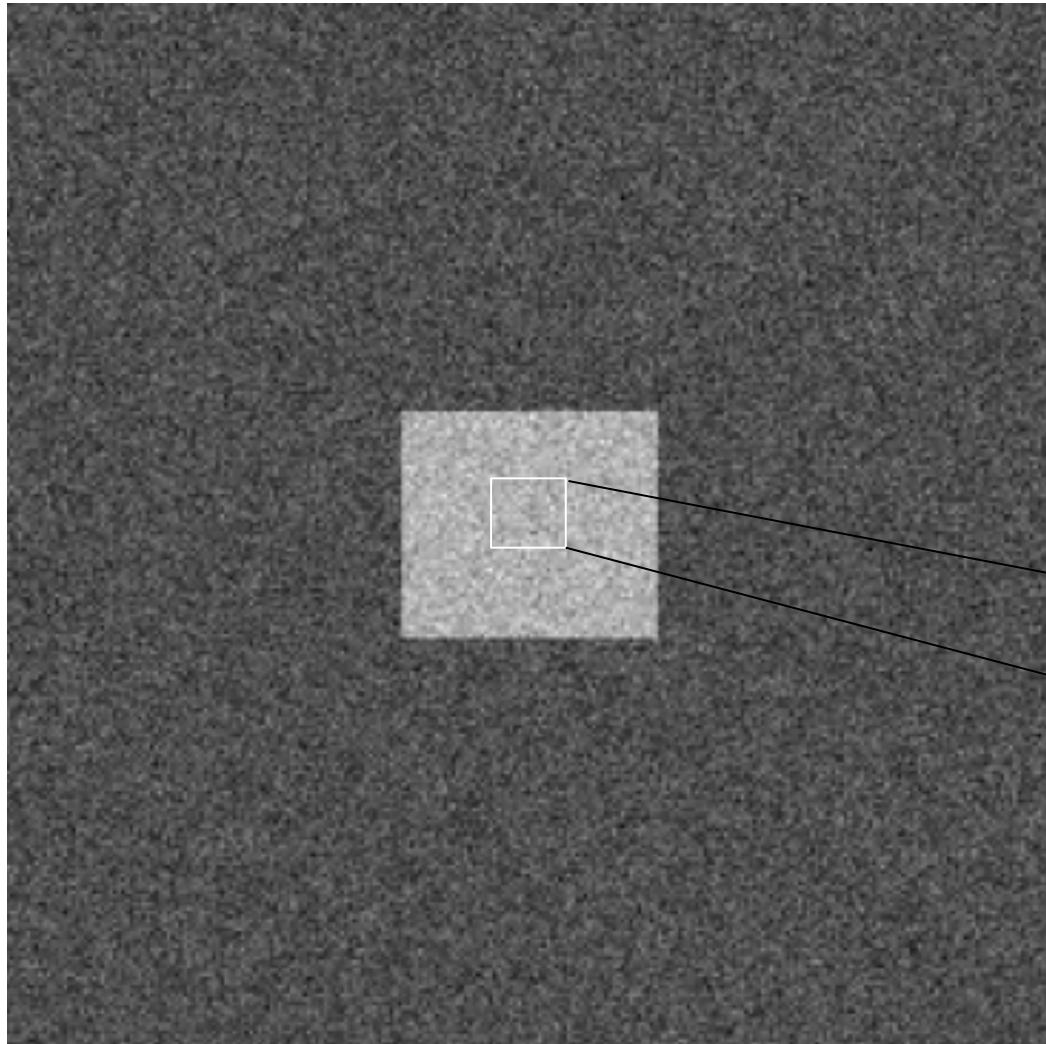


$$\text{SNR} = \frac{\text{mean of signal}}{\text{std of noise}}$$

$$\begin{aligned}\text{SNR} &= 1 / 0.2 \\ &= 5\end{aligned}$$

Rapport signal sur bruit (SNR) en pratique

Demo 03



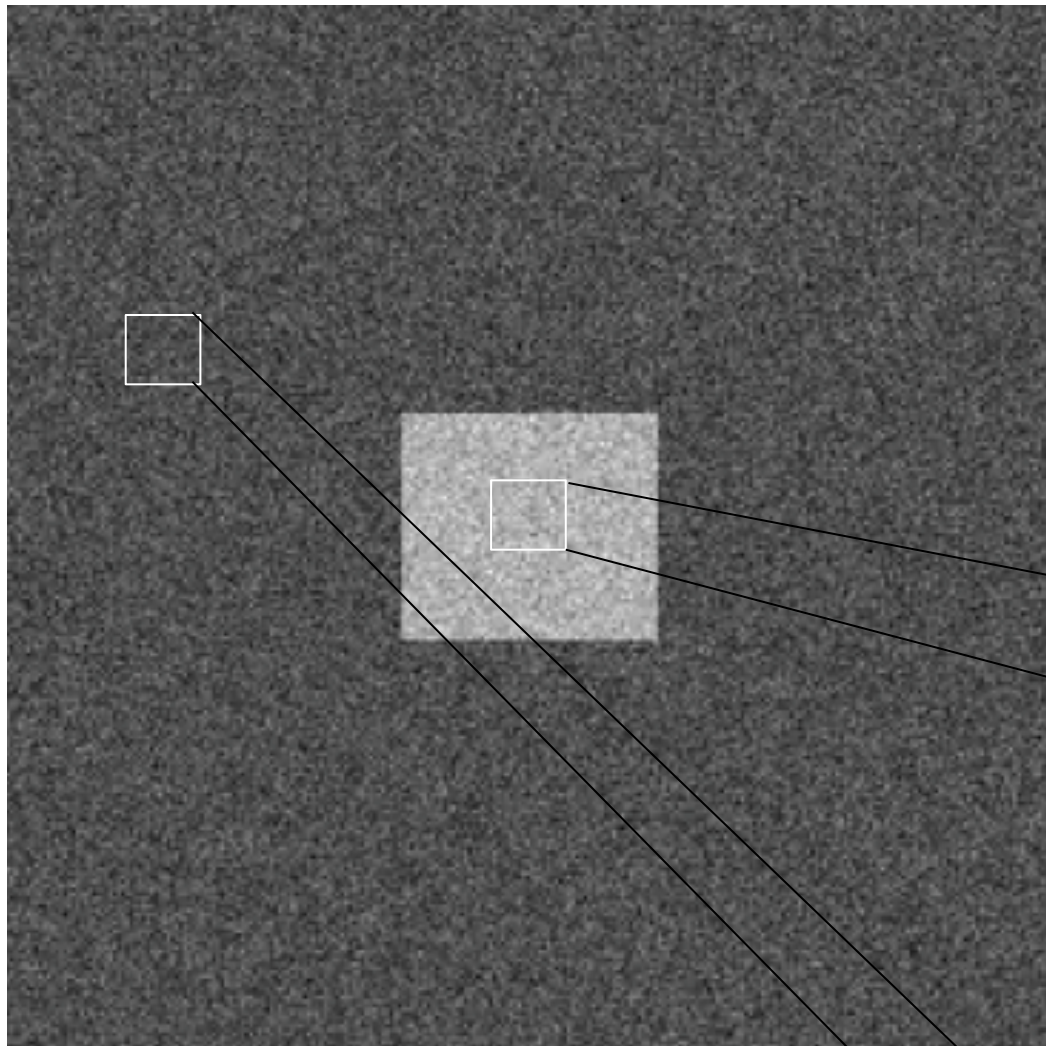
$$\text{SNR} = \frac{\text{mean of signal}}{\text{std of noise}}$$

$$\begin{aligned}\text{SNR} &= 1 / 0.2 \\ &= 5\end{aligned}$$

mean(S)

Rapport signal sur bruit (SNR) en pratique

Demo 03



$$\text{SNR} = \frac{\text{mean of signal}}{\text{std of noise}}$$

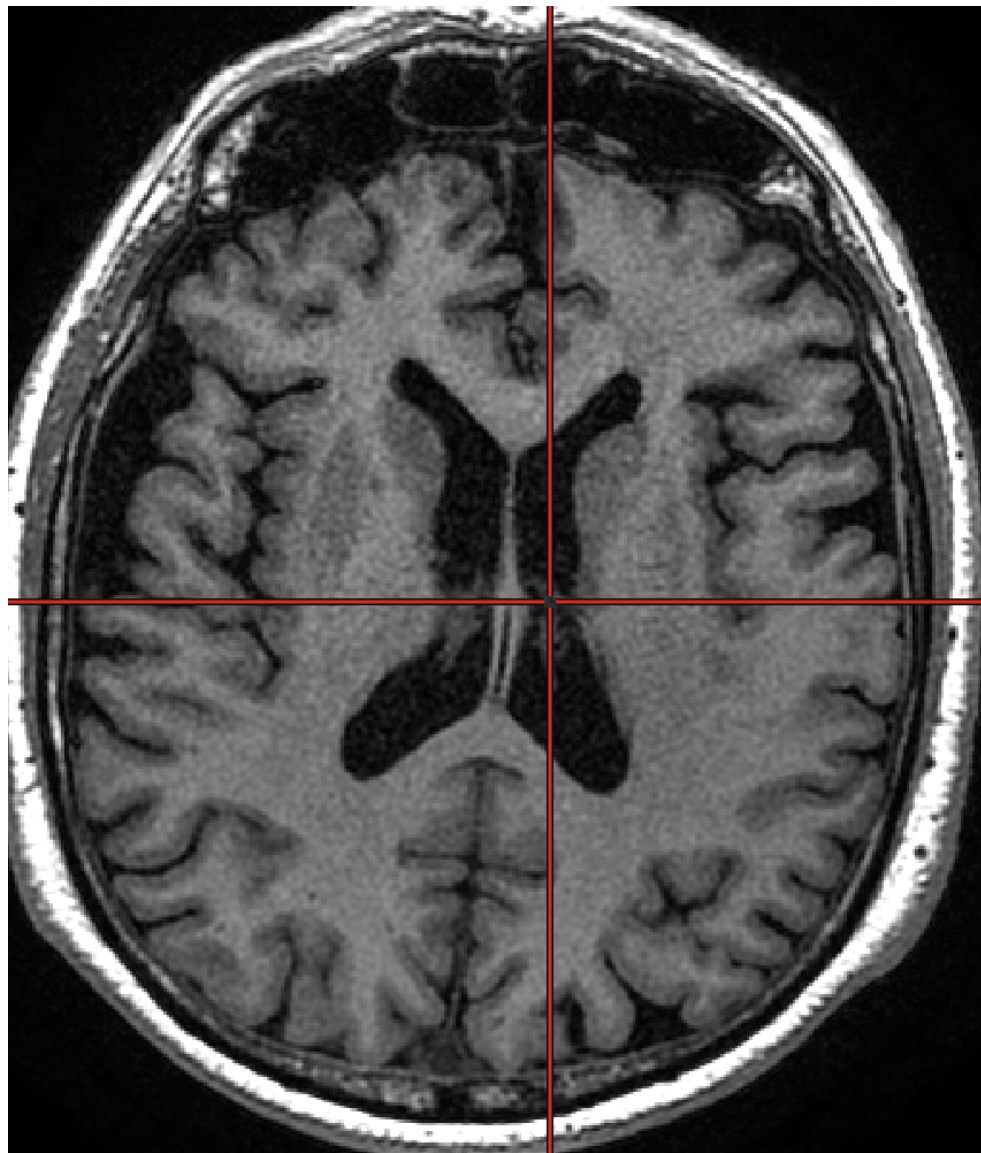
$$\begin{aligned}\text{SNR} &= 1 / 0.2 \\ &= 5\end{aligned}$$

mean(S)

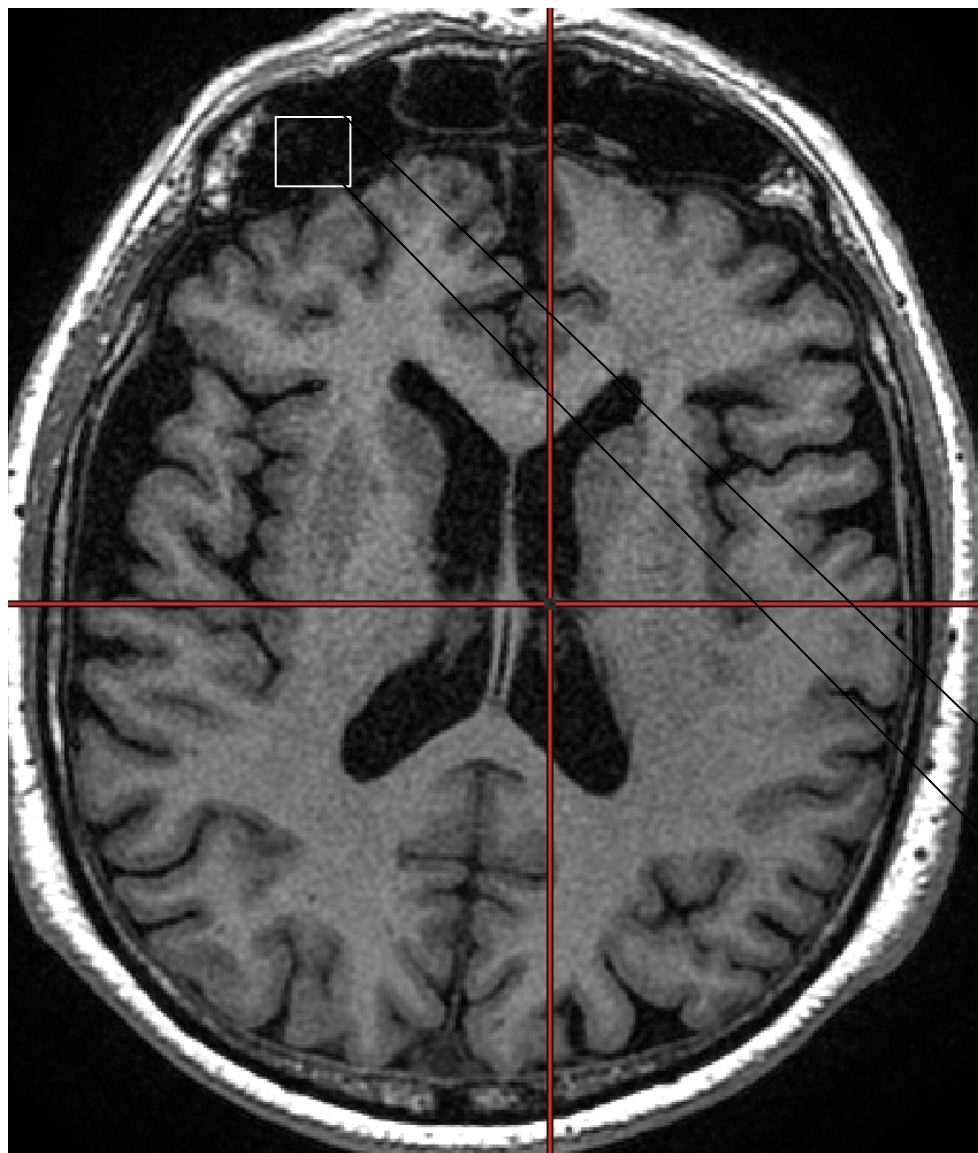
std(fond)

Estimation du SNR

Estimation du SNR

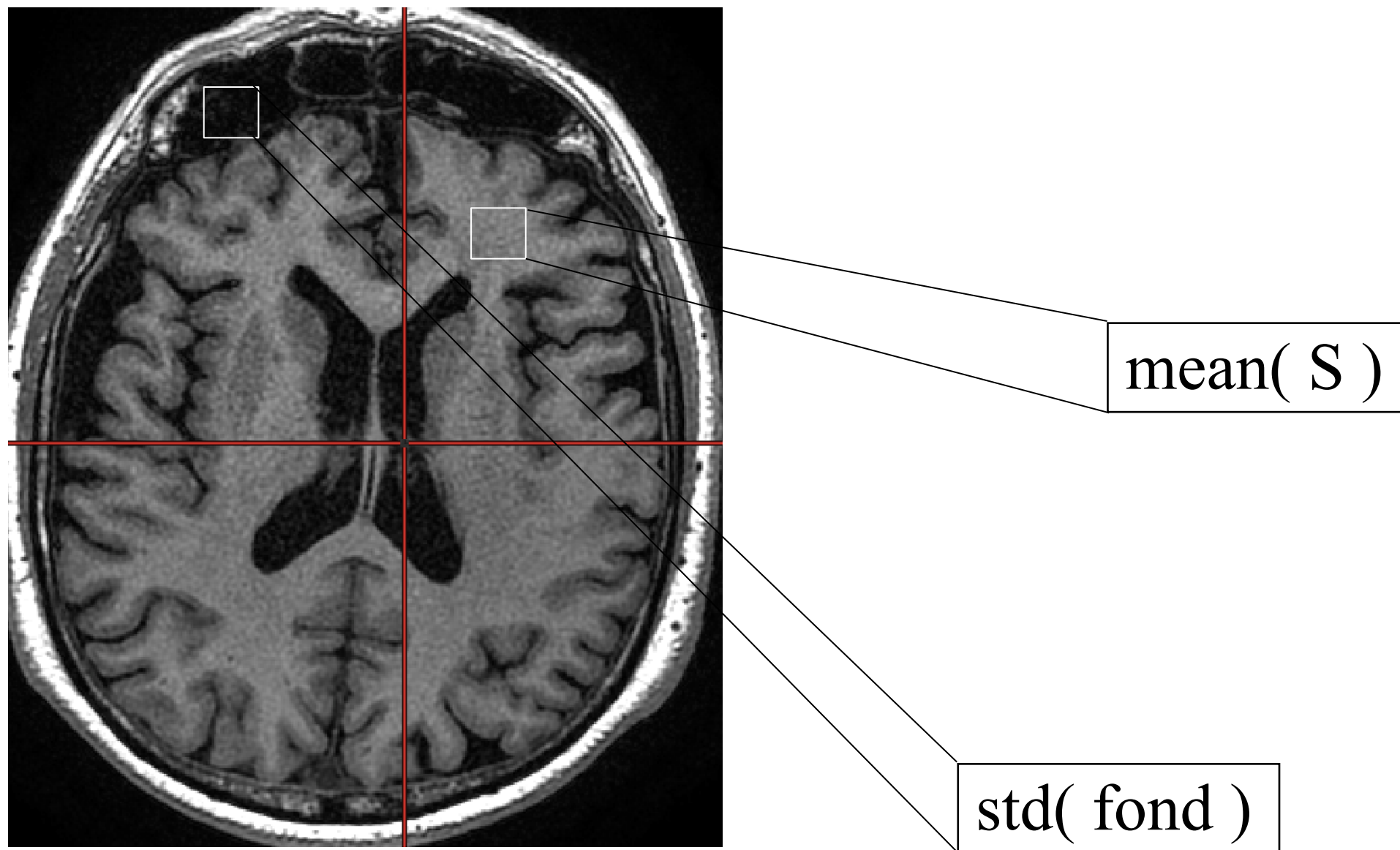


Estimation du SNR

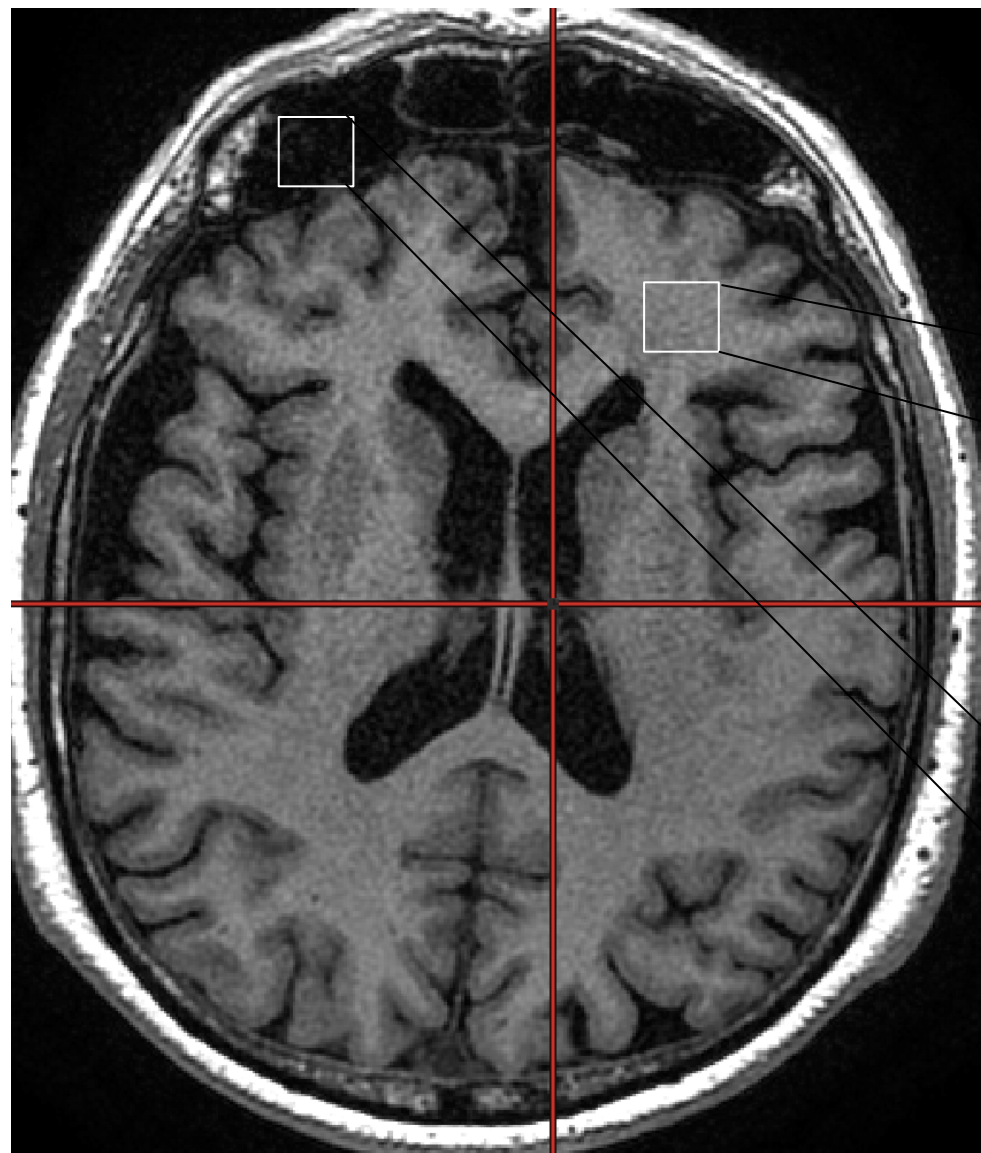


std(fond)

Estimation du SNR



Estimation du SNR



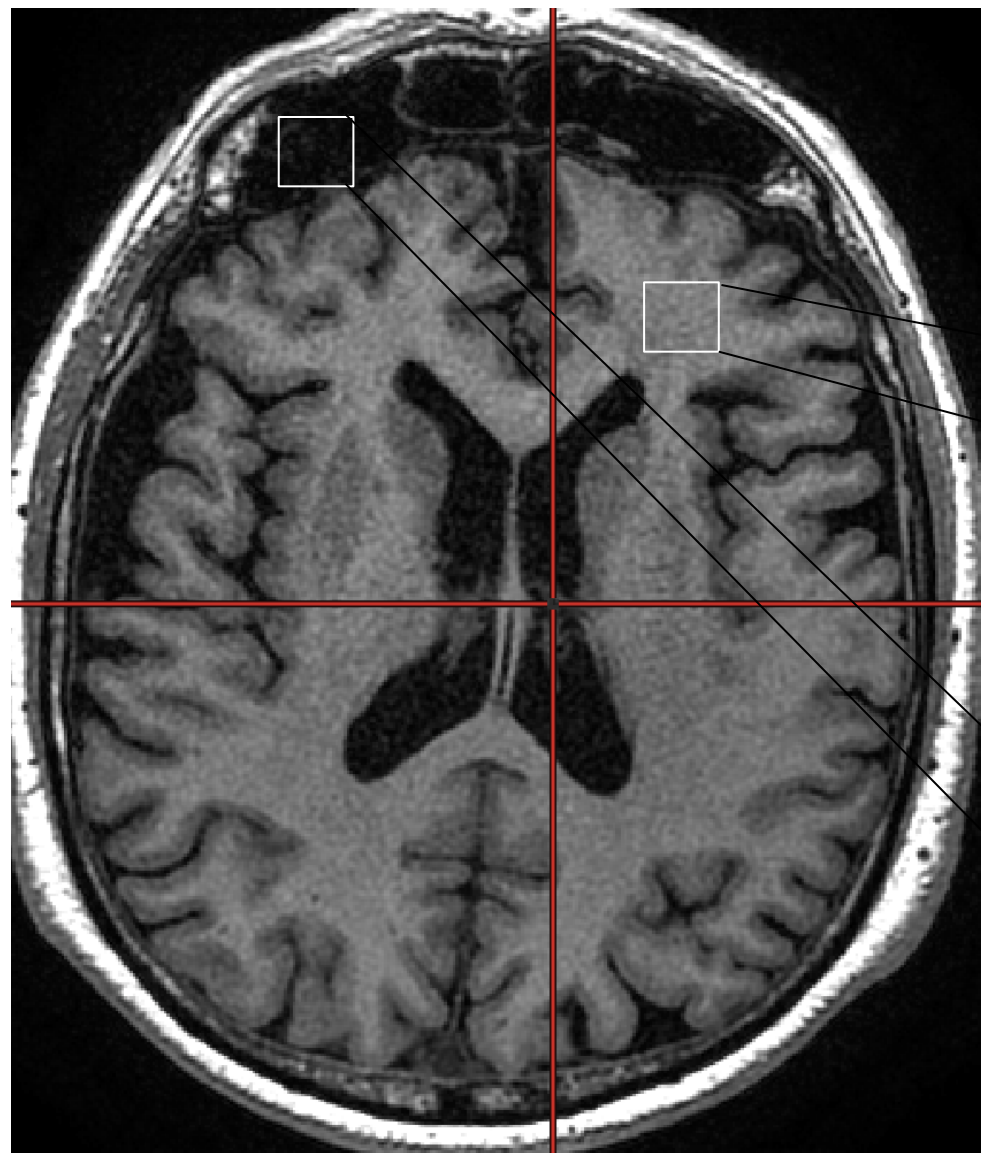
mean(S)

$$\text{SNR} = \text{mean}(S) / \text{std}(\text{fond})$$

std(fond)

Estimation du SNR

Potentiellement plusieurs
S intéressants
dans lequel il faut
rapporter le SNR



mean(S)

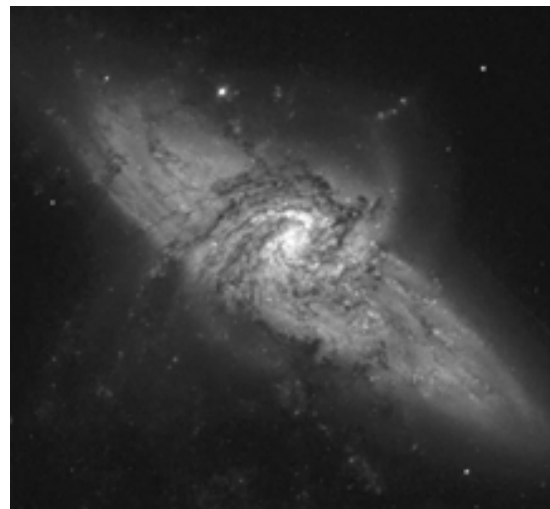
$$\text{SNR} = \text{mean}(S) / \text{std}(\text{fond})$$

std(fond)

Opérations inter-images.

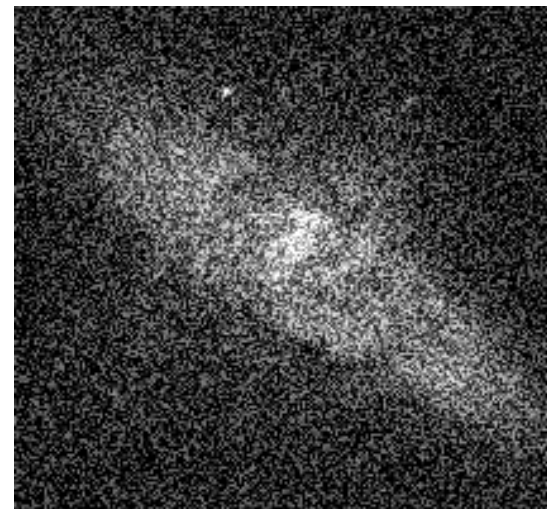
Réduction du bruit par moyennage

Image originale



$f(x,y)$

Image corrompue par
du bruit numérique



$g(x,y)$

$$g(x,y) = f(x,y) + \eta(x,y)$$

Si la lentille peut prendre plusieurs images (toujours avec un bruit non corrélé et de moyenne nulle) alors on peut réduire le bruit en additionnant ces images entre elles :

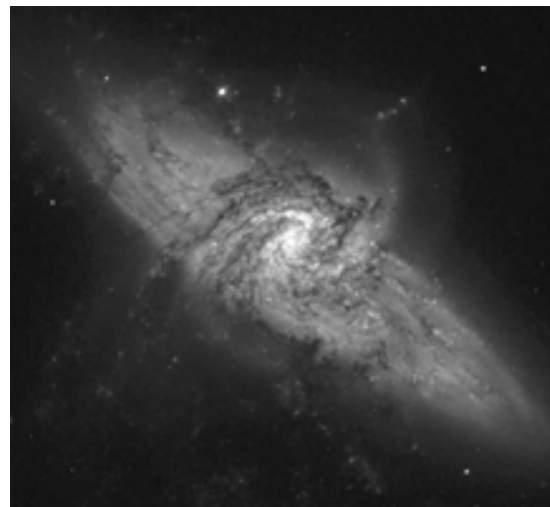
$$\bar{g}(x,y) = \frac{1}{N} \sum_{i=1}^N g_i(x,y)$$

Opérations inter-images.

TP 02

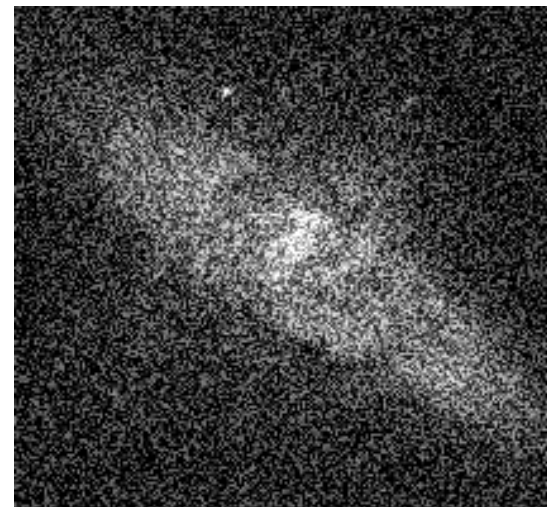
Réduction du bruit par moyennage

Image originale



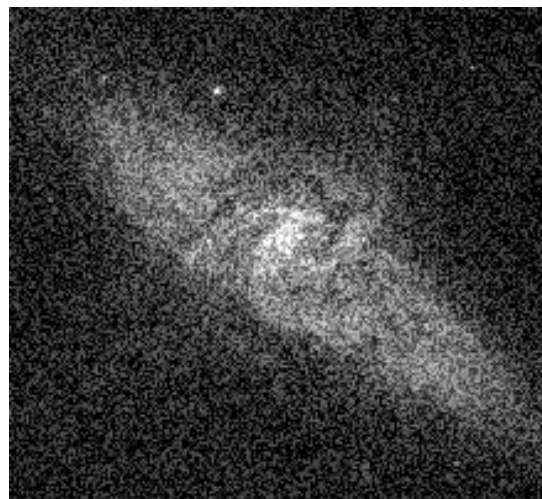
$f(x,y)$

Image corrompue par
du bruit numérique

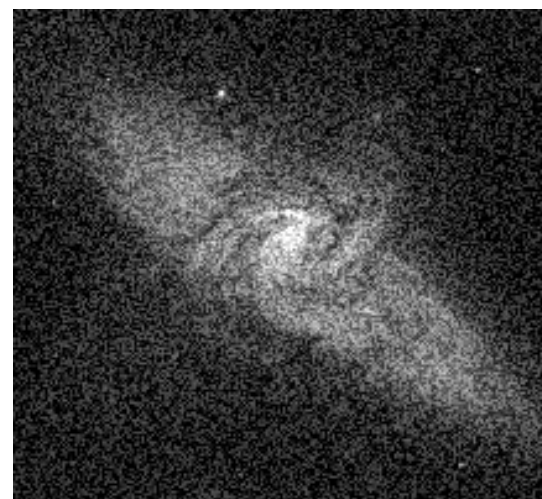


$g(x,y)$

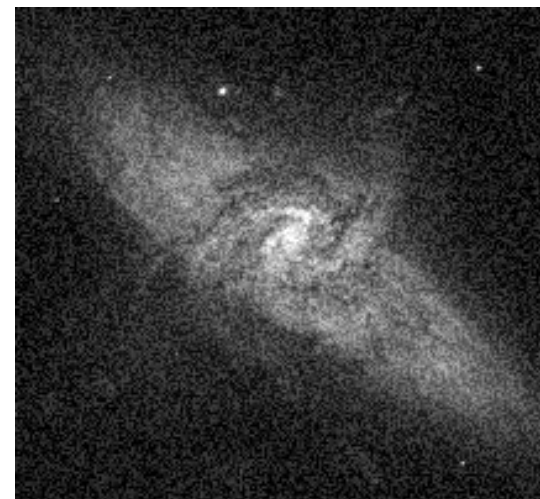
Gonzalez-Woods



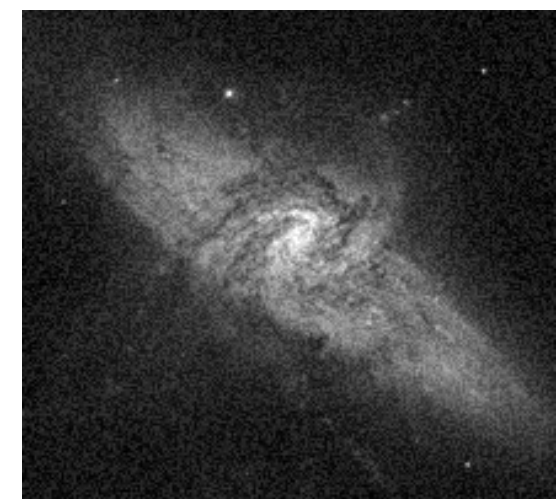
$N=2$



$N=4$



$N=8$



$N=16$

Ratio signal sur bruit : $\left(\frac{\bar{g}}{\eta}\right) = \sqrt{N} \left(\frac{g}{\eta}\right)$. Donc, pour $N=64$, le bruit dans \bar{g} est réduit d'un facteur 8.

Moyenner plusieurs image ensemble

On peut augmenter le ratio signal sur bruit d'une image en moyennant plusieurs images $g_i(x,y)$ dont **le contenu est toujours le même** ($f(x,y)$) plus ou moins du **bruit blanc additif**.

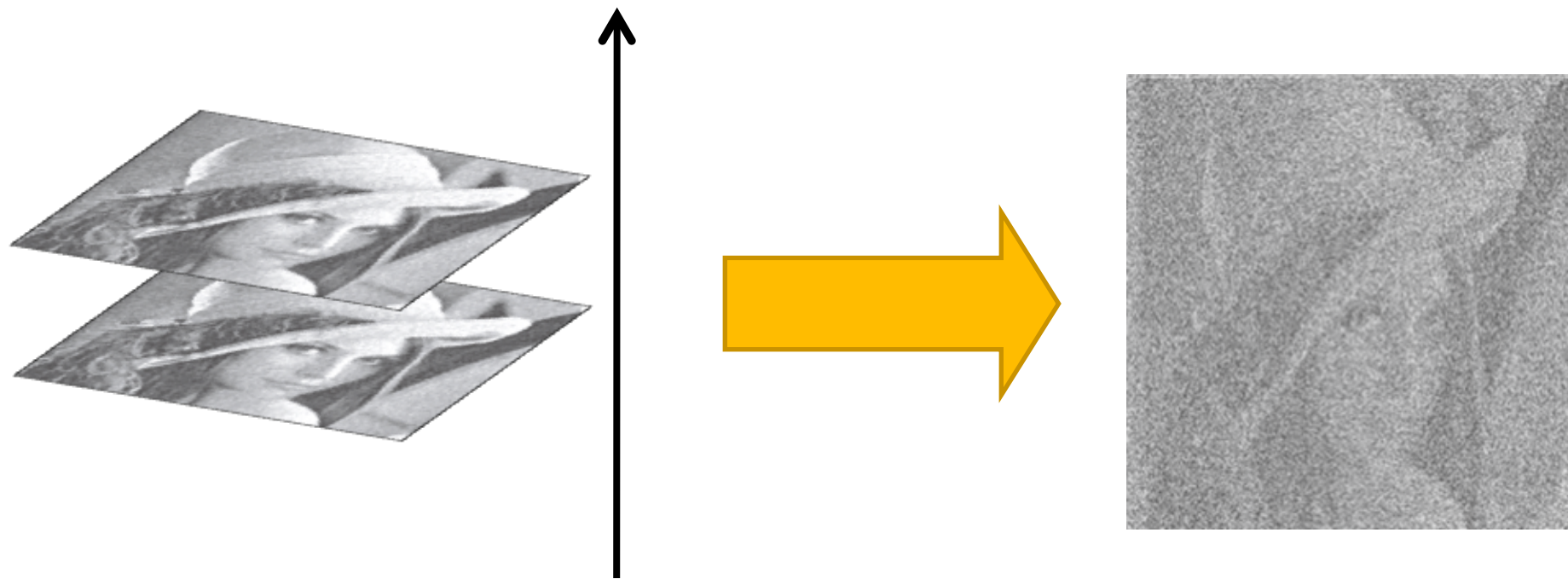
$$g_i(x,y) = f(x,y) + \eta(x,y)$$



Bruit blanc additif

Moyenner plusieurs image ensemble

On peut augmenter le ratio signal sur bruit d'une image en moyennant plusieurs images $g_i(x,y)$ dont **le contenu est toujours le même** ($f(x,y)$) plus ou moins du **bruit blanc additif**.

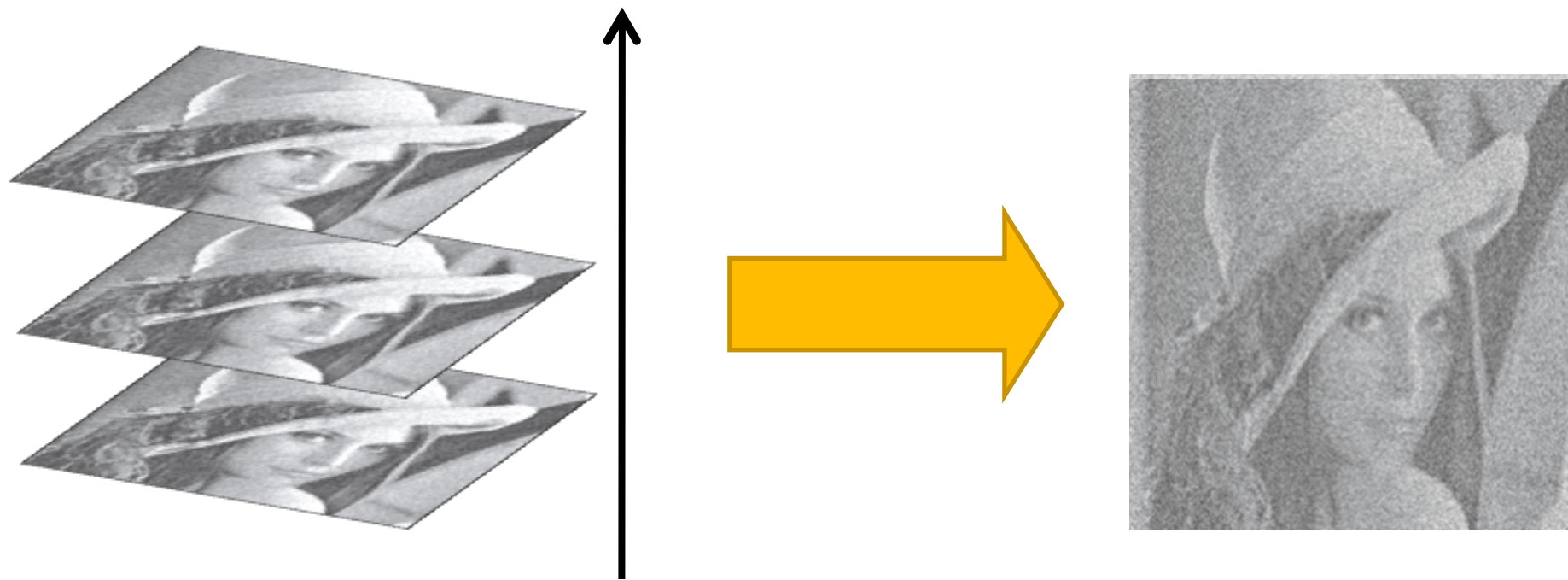


$$g_i(x,y) = f(x,y) + \eta(x,y)$$

Bruit blanc additif

Moyenner plusieurs image ensemble

On peut augmenter le ratio signal sur bruit d'une image en moyennant plusieurs images $g_i(x,y)$ dont **le contenu est toujours le même** ($f(x,y)$) plus ou moins du **bruit blanc additif**.

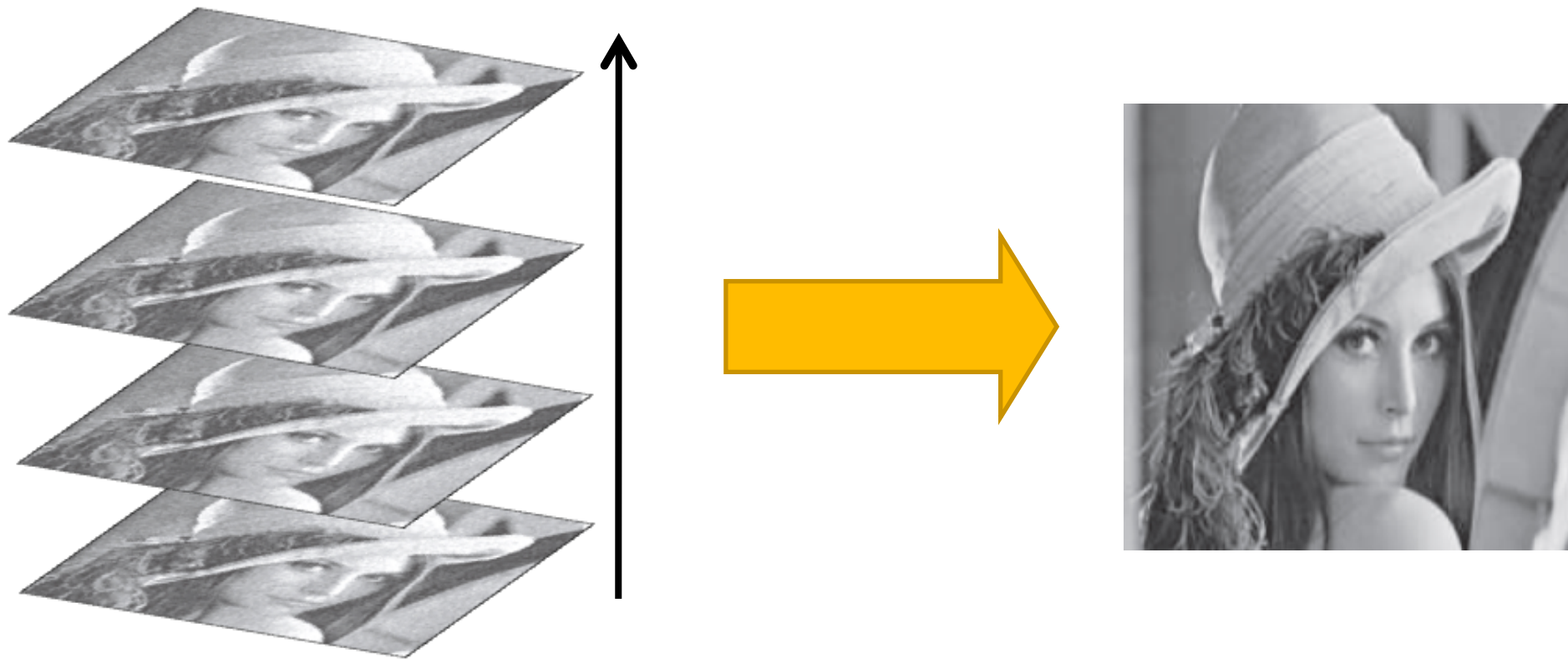


$$g_i(x,y) = f(x,y) + \eta(x,y)$$

Bruit blanc additif

Moyenner plusieurs image ensemble

On peut augmenter le ratio signal sur bruit d'une image en moyennant plusieurs images $g_i(x,y)$ dont **le contenu est toujours le même** ($f(x,y)$) plus ou moins du **bruit blanc additif**.

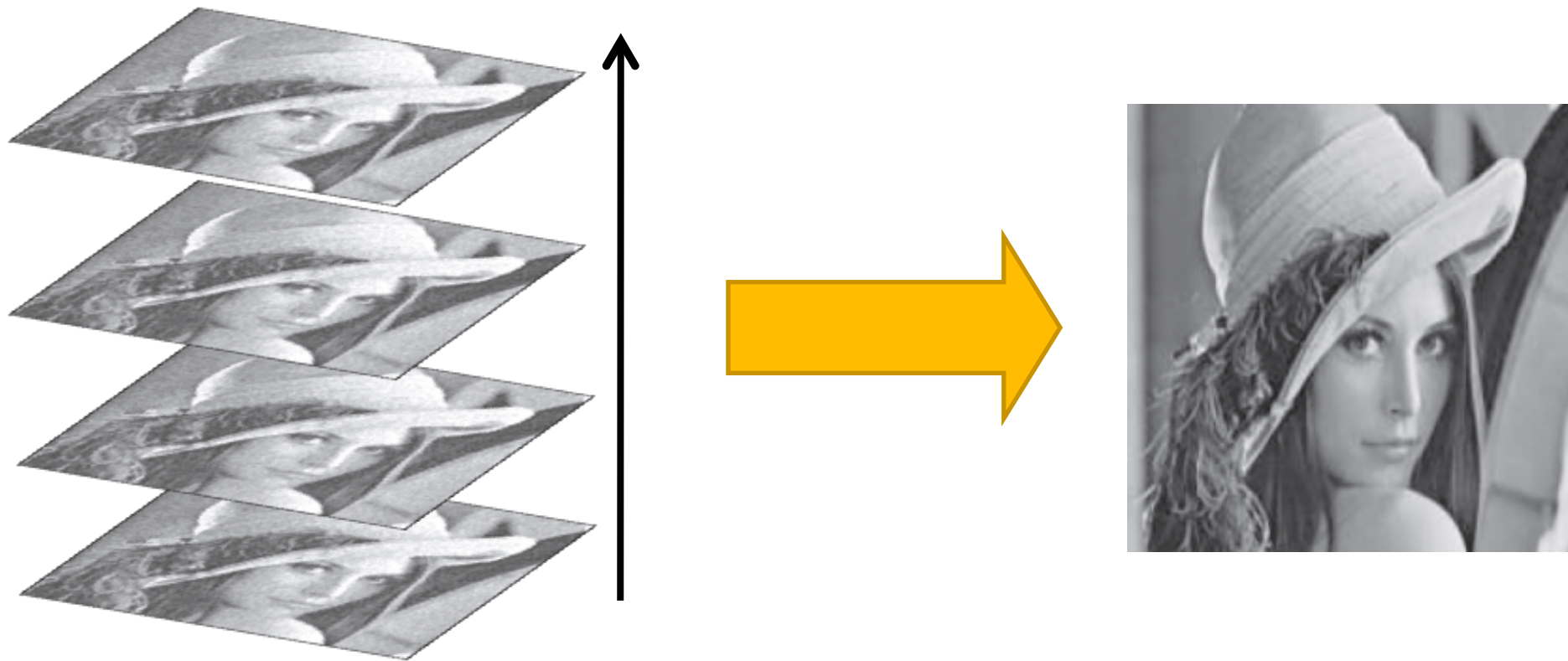


$$g_i(x,y) = f(x,y) + \eta(x,y)$$

Bruit blanc additif

Moyenner plusieurs image ensemble

On peut augmenter le ratio signal sur bruit d'une image en moyennant plusieurs images $g_i(x,y)$ dont le **contenu est toujours le même** ($f(x,y)$) plus ou moins du **bruit blanc additif**.



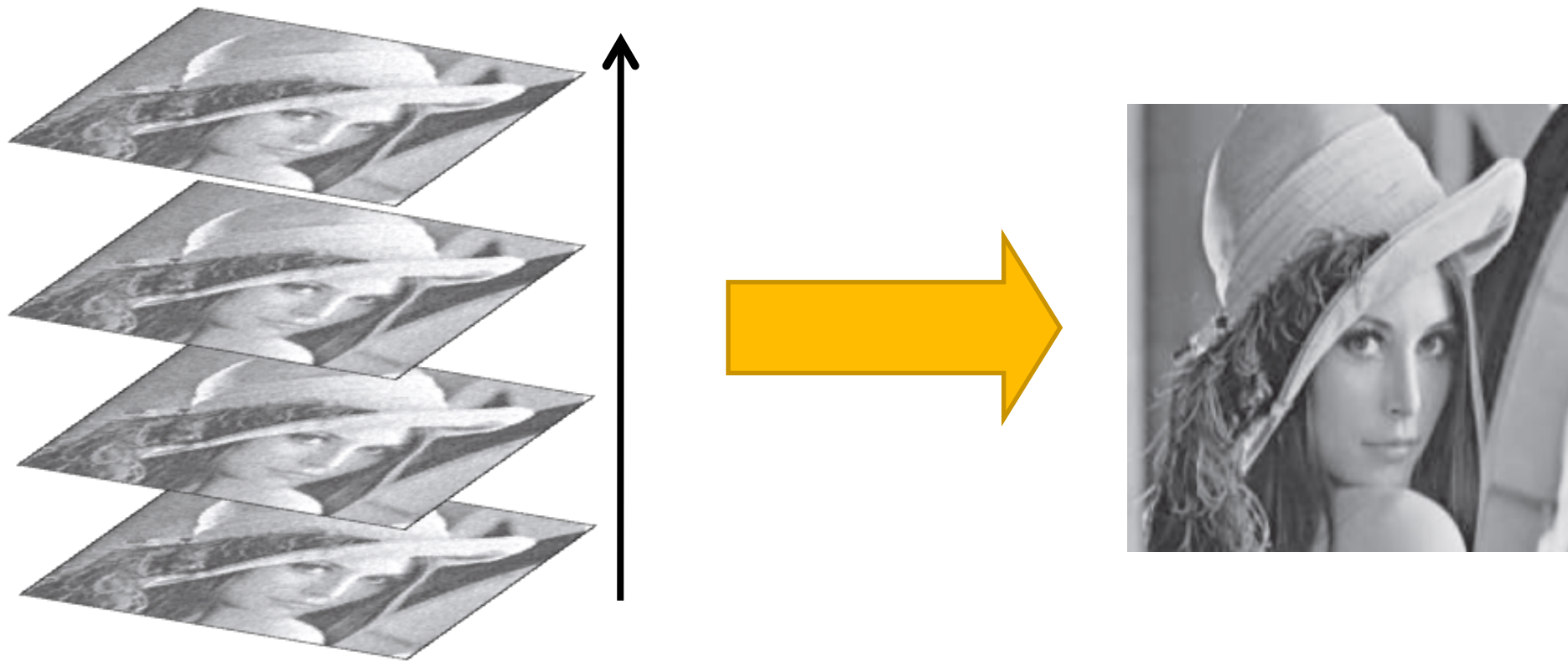
$$g_i(x,y) = f(x,y) + \eta(x,y)$$

↑
Bruit blanc additif

N moyennes augmente le SNR
d'un facteur?

Moyenner plusieurs image ensemble

On peut augmenter le ratio signal sur bruit d'une image en moyennant plusieurs images $g_i(x,y)$ dont le **contenu est toujours le même** ($f(x,y)$) plus ou moins du **bruit blanc additif**.



$$g_i(x,y) = f(x,y) + \eta(x,y)$$

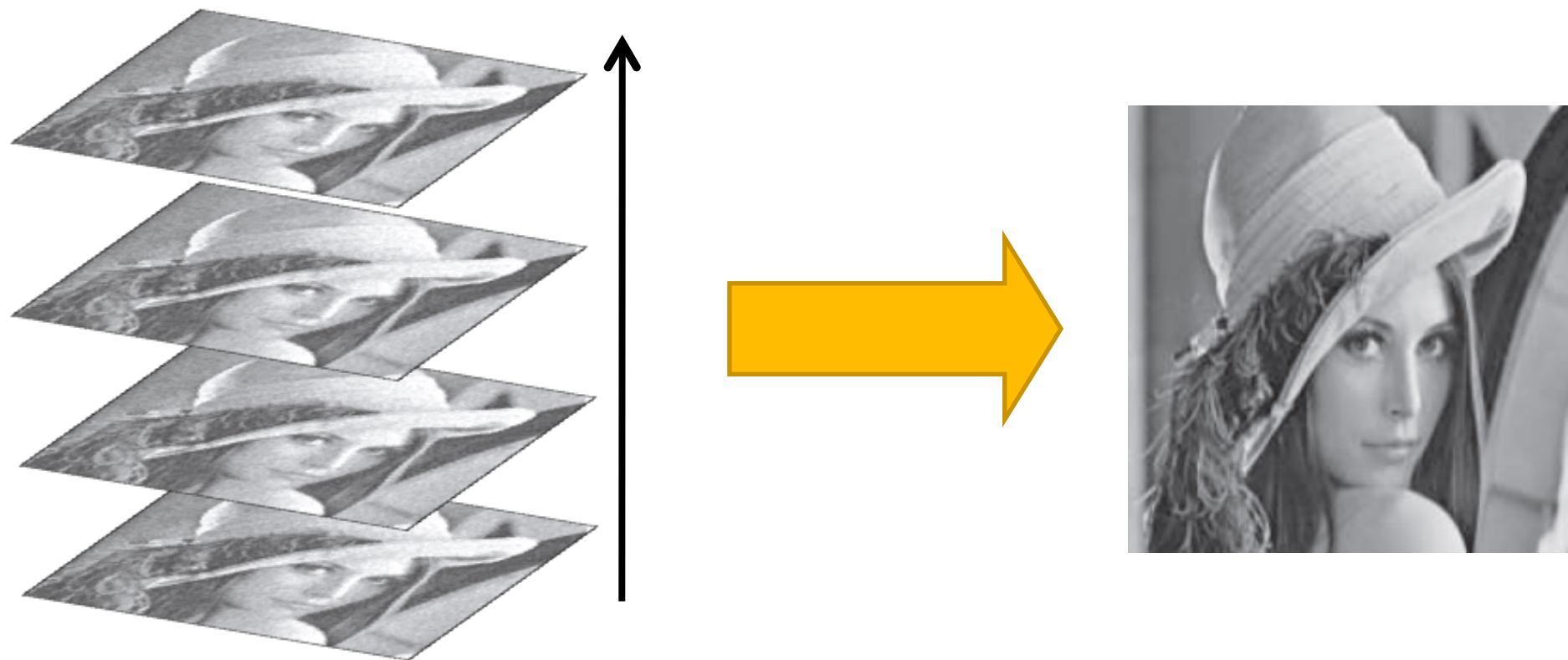
↑
Bruit blanc additif

N moyennes augmente le SNR
d'un facteur?

$$\sqrt{N}$$

Moyenner plusieurs image ensemble

On peut augmenter le ratio signal sur bruit d'une image en moyennant plusieurs images $g_i(x,y)$ dont le **contenu est toujours le même** ($f(x,y)$) plus ou moins du **bruit blanc additif**.



$$g_i(x,y) = f(x,y) + \eta(x,y)$$

↑
Bruit blanc additif

N moyennes augmente le SNR
d'un facteur? \sqrt{N}

Moyenner plusieurs image ensemble

On peut augmenter le ratio signal sur bruit d'une image en moyennant plusieurs images $g_i(x,y)$ dont **le contenu est toujours le même** ($f(x,y)$) plus ou moins du **bruit blanc additif**.

Reasoning

- Let the clean signal be constant with mean μ_s .
- Each noisy image has noise with standard deviation σ_n .
- When you average N independent noisy images:

$$\text{average noise std} = \frac{\sigma_n}{\sqrt{N}}$$

because the variance of the mean of N independent samples is reduced by N , so the standard deviation reduces by \sqrt{N} .

- The signal mean stays the same (μ_s).
- So:

$$\text{SNR}_{\text{avg}} = \frac{\mu_s}{\sigma_n / \sqrt{N}} = \sqrt{N} \cdot \frac{\mu_s}{\sigma_n} = \sqrt{N} \cdot \text{SNR}_{\text{single}}$$

Moyenner plusieurs image ensemble

On peut augmenter le ratio signal sur bruit d'une image en moyennant plusieurs images $g_i(x,y)$ dont **le contenu est toujours le même** ($f(x,y)$) plus ou moins du **bruit blanc additif**.

Reasoning

- Let the clean signal be constant with mean μ_s .
- Each noisy image has noise with standard deviation σ_n .
- When you average N independent noisy images:

$$\text{average noise std} = \frac{\sigma_n}{\sqrt{N}}$$

because the variance of the mean of N independent samples is reduced by N , so the standard deviation reduces by \sqrt{N} .

- The signal mean stays the same (μ_s).
- So:

$$\text{SNR}_{\text{avg}} = \frac{\mu_s}{\sigma_n / \sqrt{N}} = \sqrt{N} \cdot \frac{\mu_s}{\sigma_n} = \sqrt{N} \cdot \text{SNR}_{\text{single}}$$

Type de bruit

Bruit

$$g(x,y) = f(x,y) + \eta(x,y)$$



Bruit additif non corrélé

Espace de Fourier

Bruit

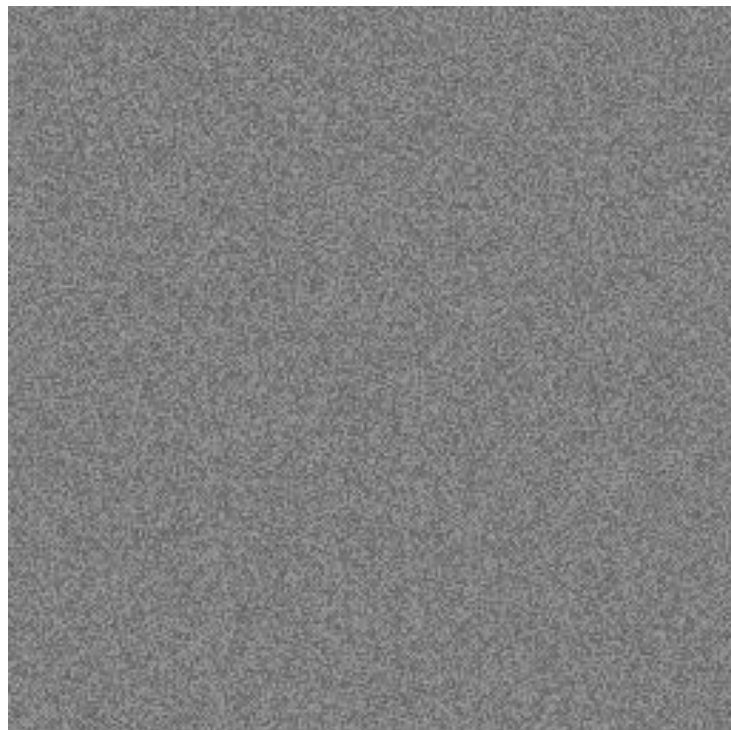
$$g(x,y) = f(x,y) + \eta(x,y)$$



Bruit additif non corrélé

En général, **le bruit est blanc**, c-à-d que sa distribution fréquentielle est globalement uniforme

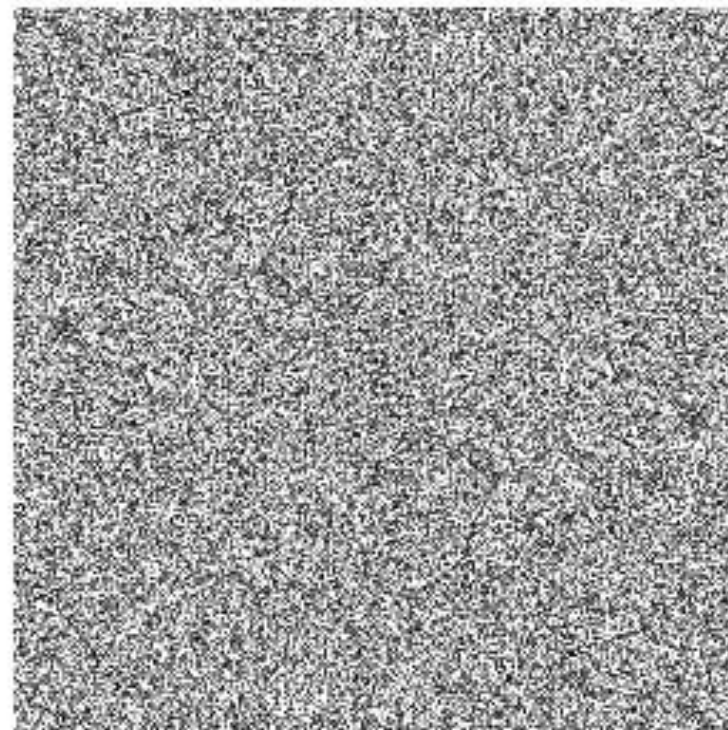
Bruit blanc (η)



\mathcal{F}



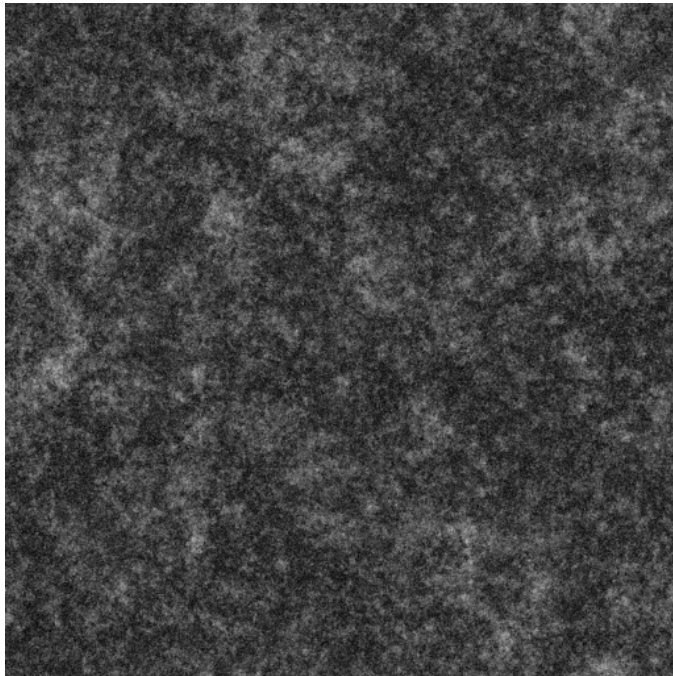
Espace de Fourier



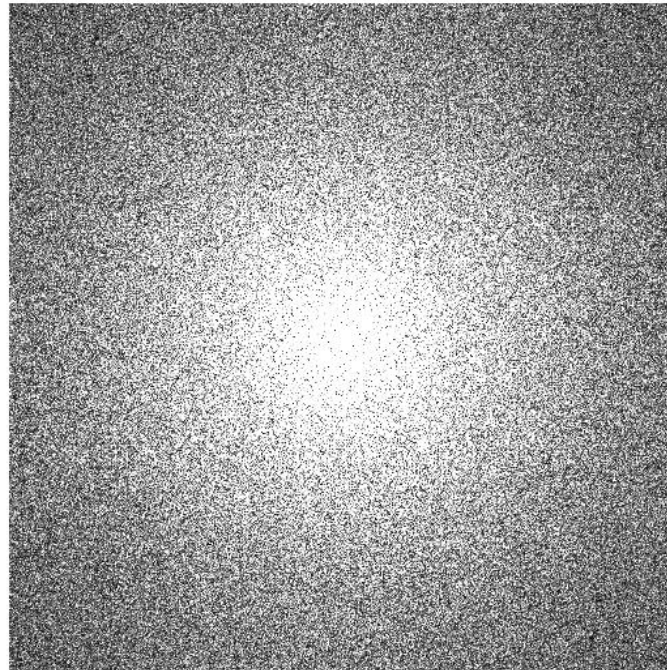
Bruit

Mais il existe plusieurs autres types de bruits additifs

Bruit rose



Espace de Fourier



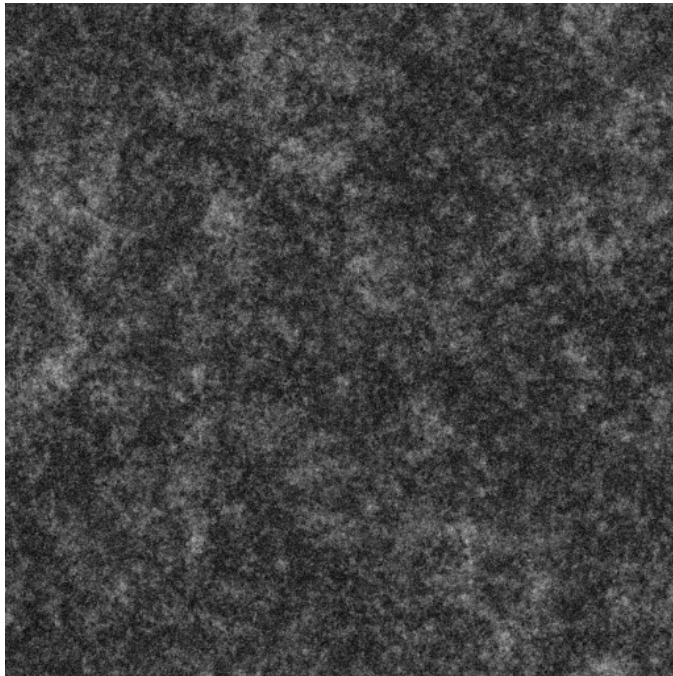
\mathcal{F}



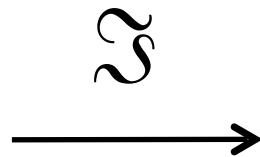
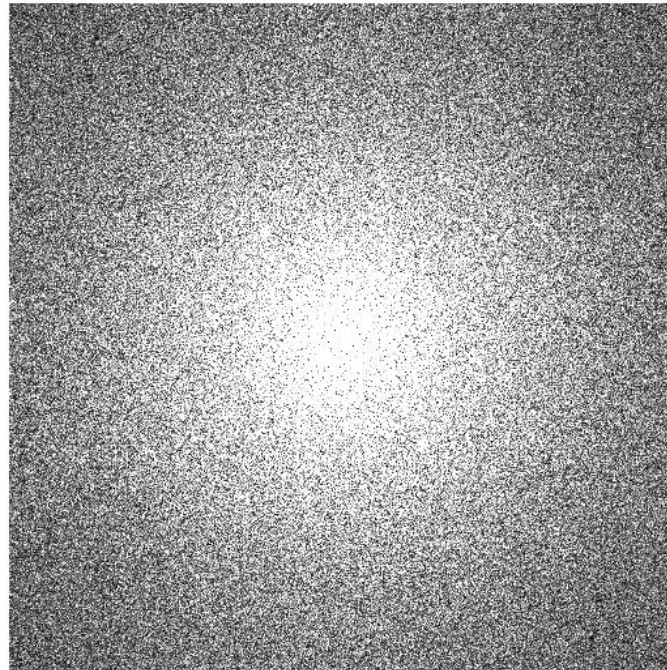
Bruit

Mais il existe plusieurs autres types de bruits additifs

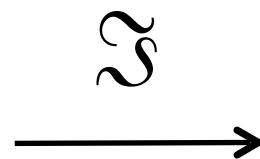
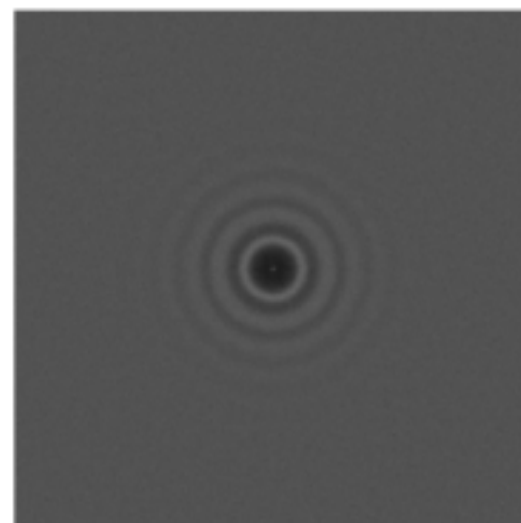
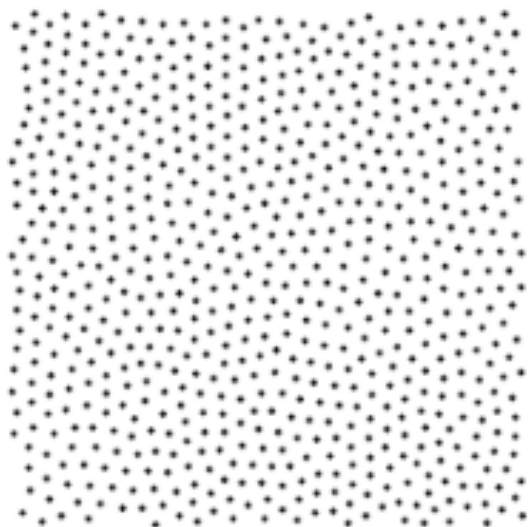
Bruit rose



Espace de Fourier



Bruit bleu



Exemple bruit bleu



Bruit

La « **couleur** » du bruit indique sa distribution spectrale.

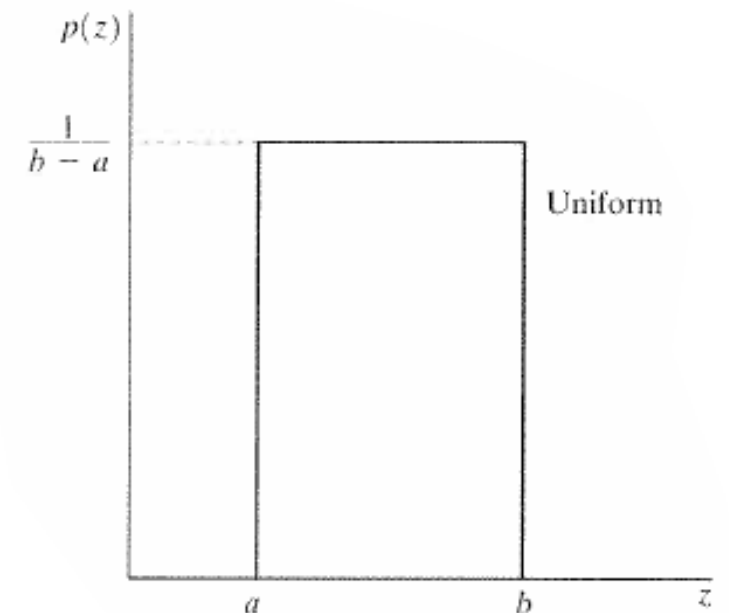
Demo 03

Pour une « couleur » de bruit, il peut exister plusieurs types de distributions spatiales.

Une variable de bruit « z » est considérée comme étant une **variable aléatoire** suivant une **densité de probabilités**

Bruit uniforme

$$p(z) = \frac{1}{b-a} \quad \forall a \leq z \leq b$$



Bruit

La « **couleur** » du bruit indique sa distribution spectrale.

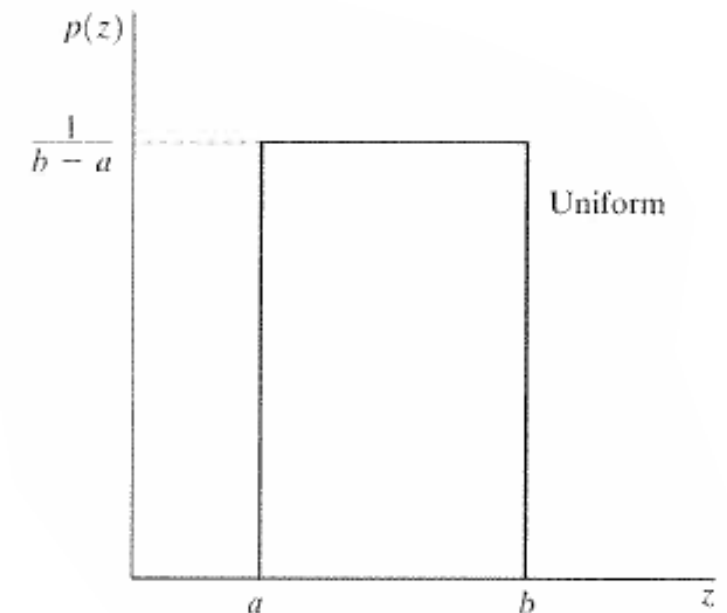
Demo 03

Pour une « couleur » de bruit, il peut exister plusieurs types de distributions spatiales.

Une variable de bruit « z » est considérée comme étant une **variable aléatoire** suivant une **densité de probabilités**

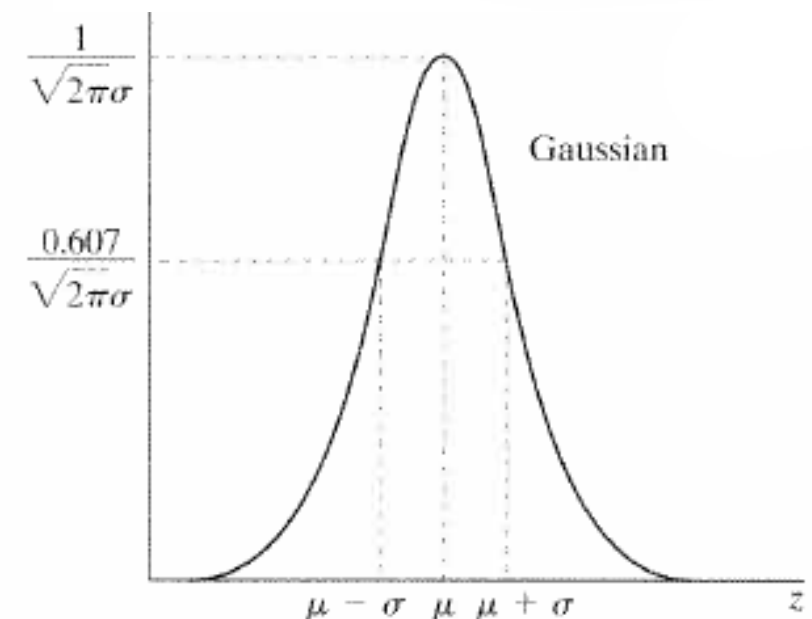
Bruit uniforme

$$p(z) = \frac{1}{b-a} \quad \forall a \leq z \leq b$$



Bruit gaussien

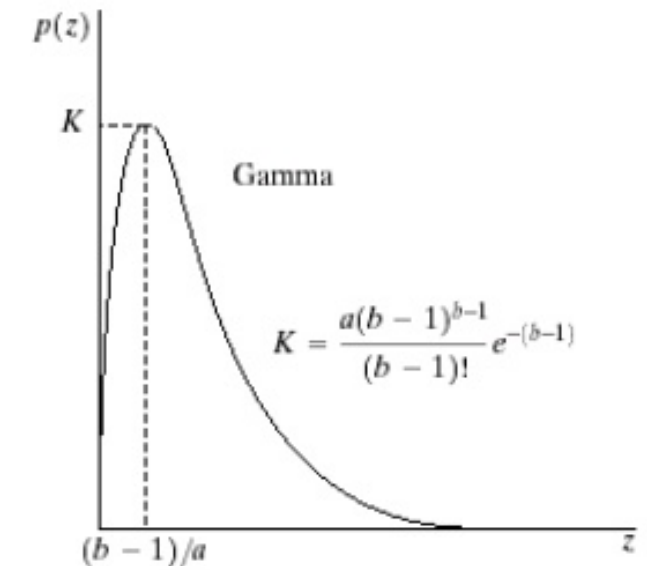
$$p(z) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(z-\mu)^2}{2\sigma^2}}$$



Bruit

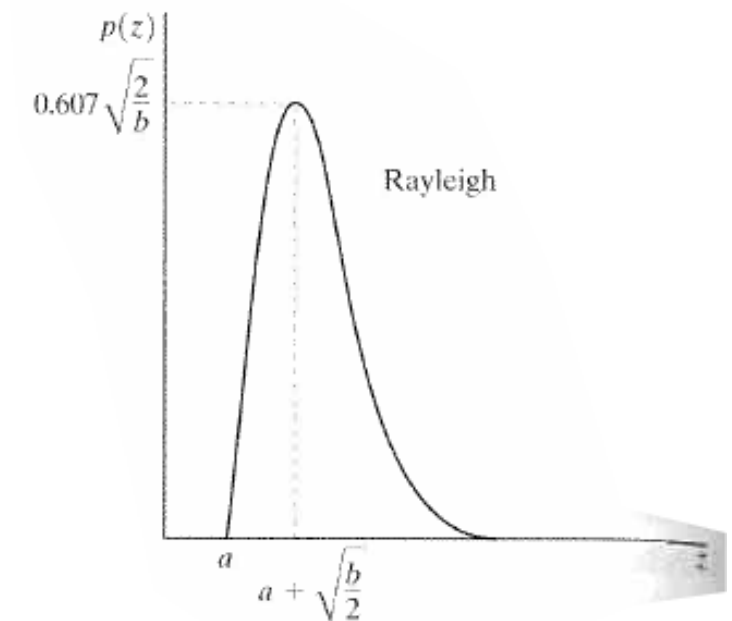
Bruit gamma

$$p(z) = \frac{a^b z^{b-1}}{(b-1)!} e^{-az} \quad \forall z \geq 0$$



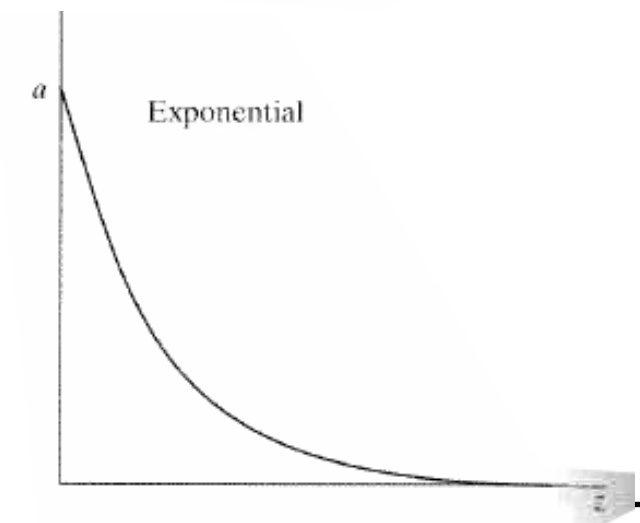
Bruit de Rayleigh

$$p(z) = \frac{2(z-a)}{b} e^{-\frac{(z-a)^2}{b}} \quad \forall z \geq a$$



Bruit exponentiel

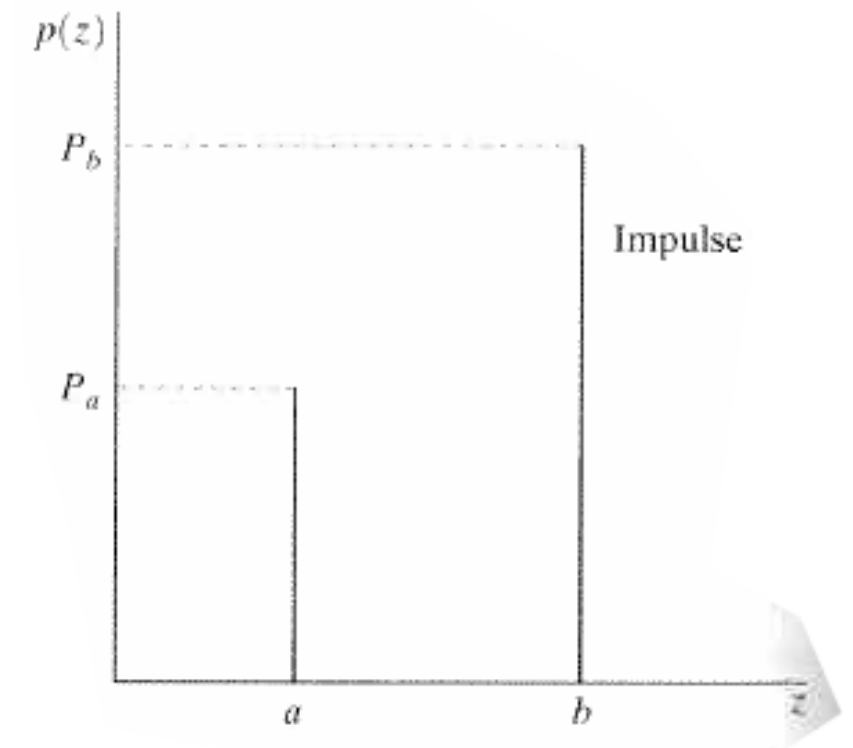
$$p(z) = a e^{-az} \quad \forall z \geq 0$$



Bruit

Bruit poivre et sel

$$p(z) = \frac{1}{b-a} \quad \forall a \leq z \leq b$$



Exemple de bruit sur une image composée de 3 régions uniformes

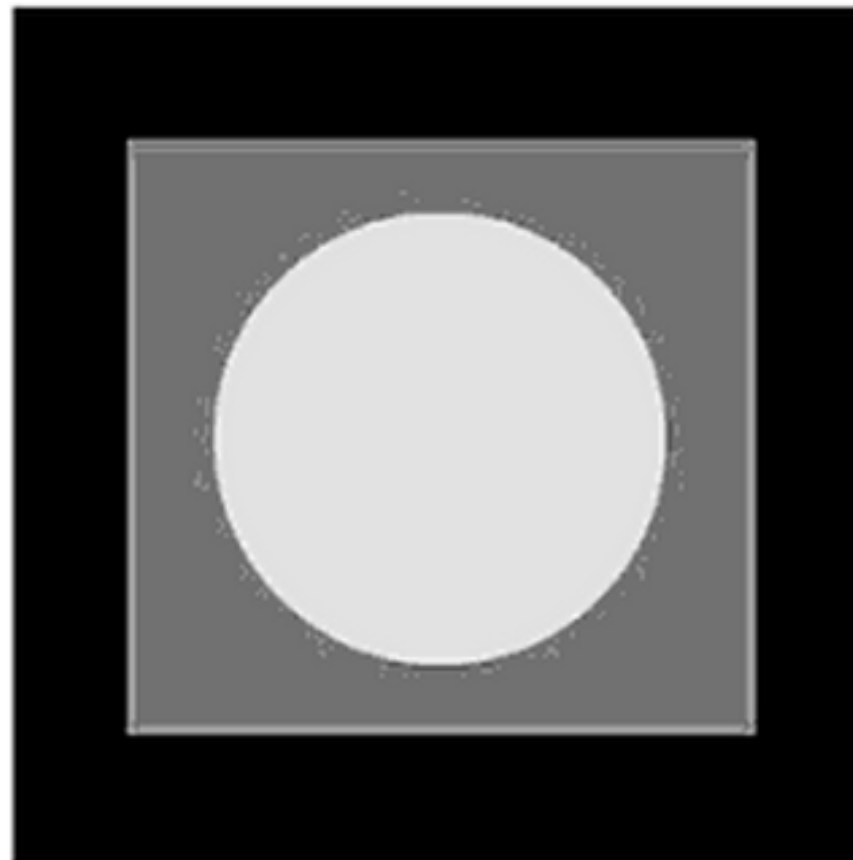
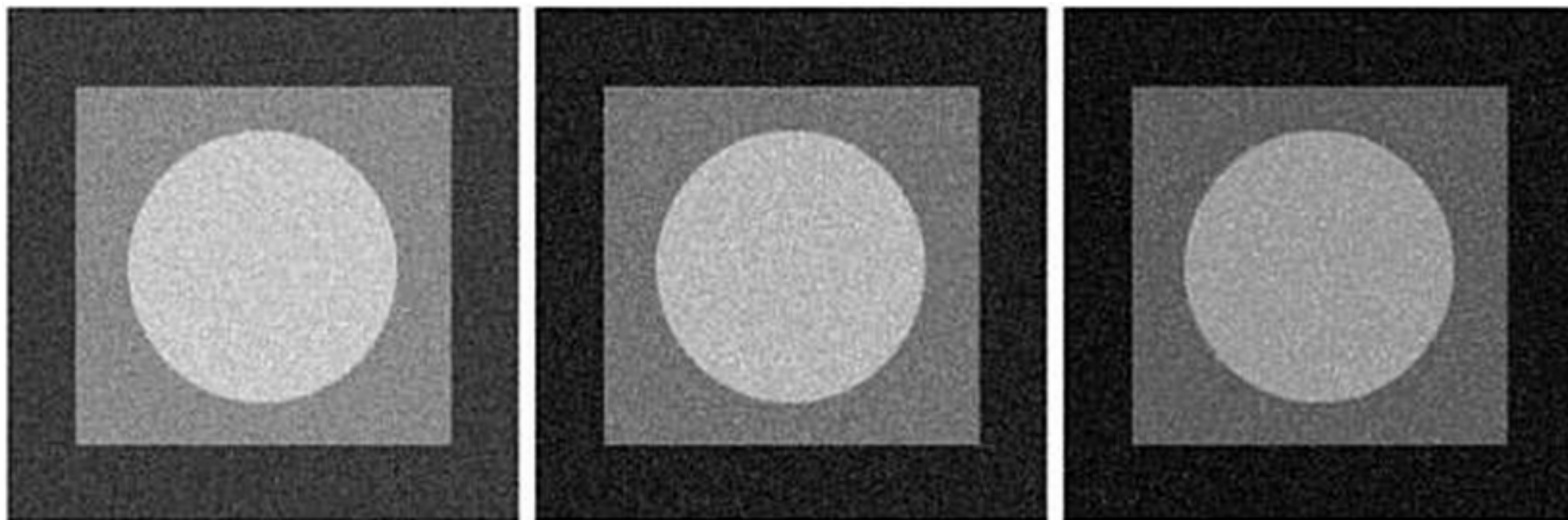
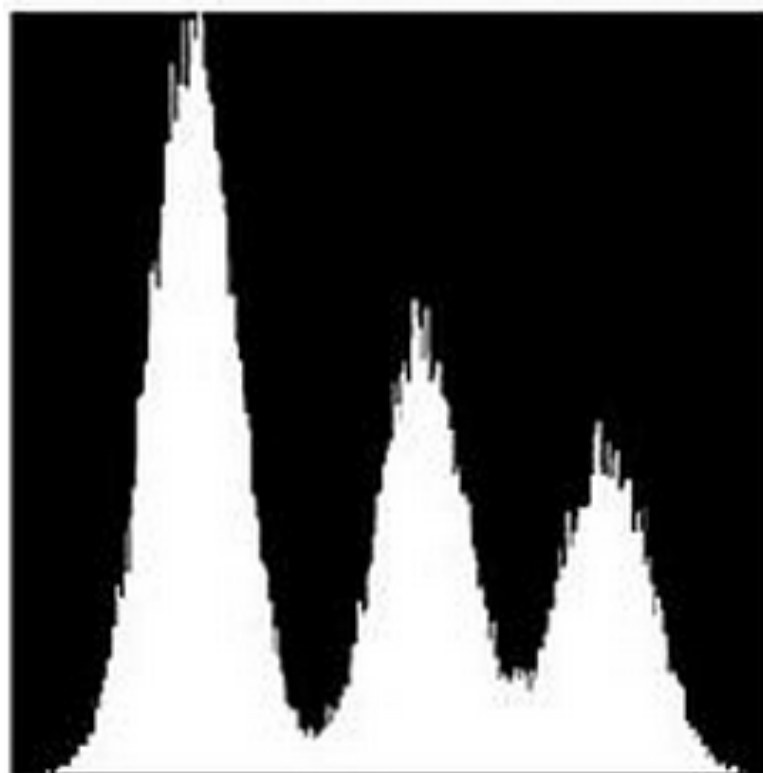
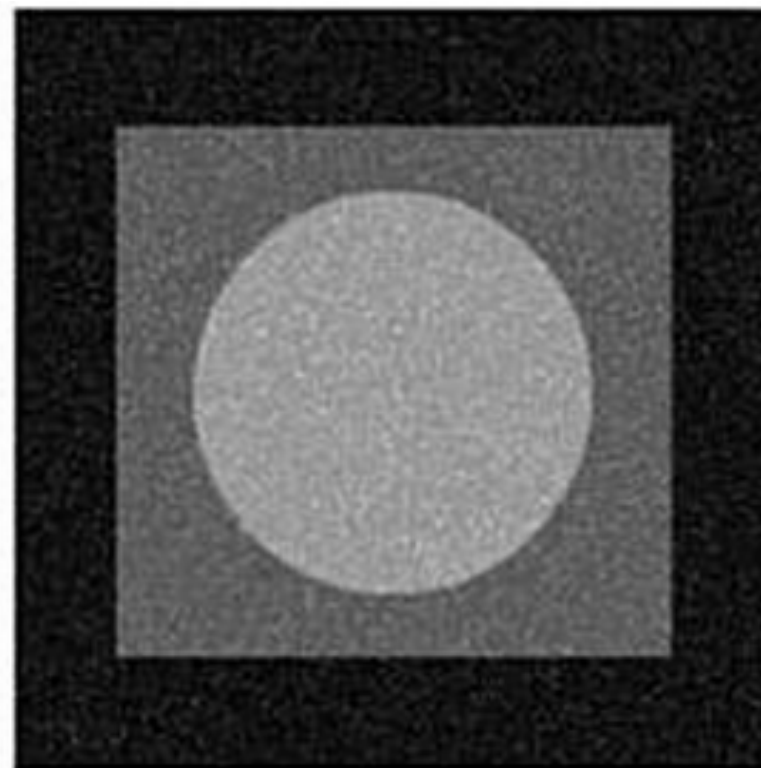
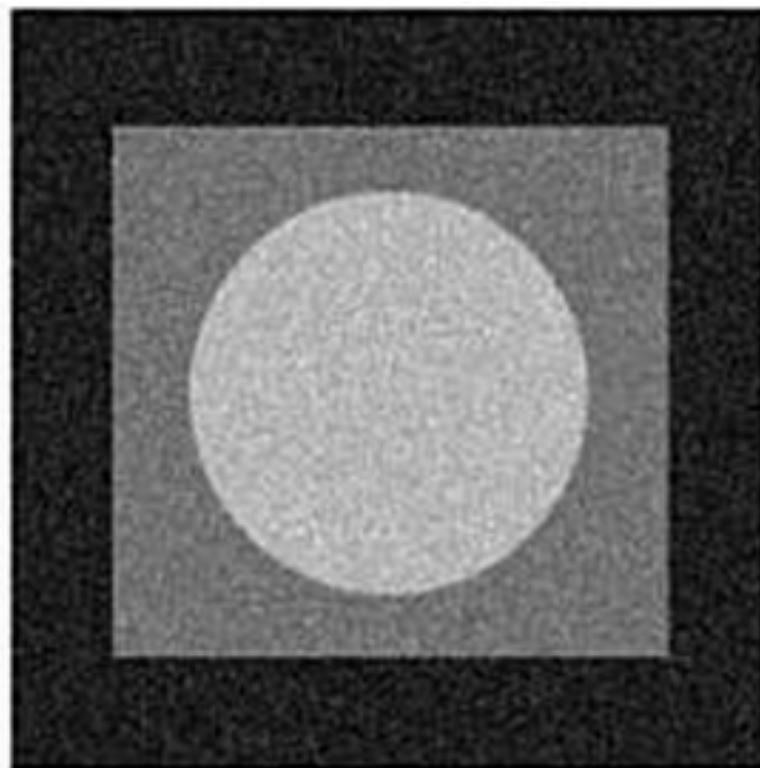
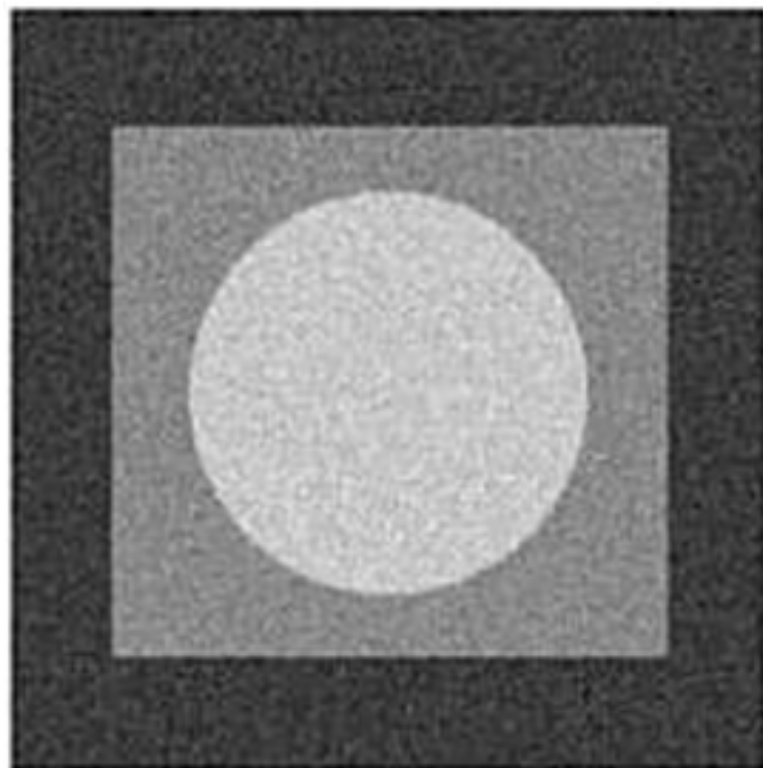
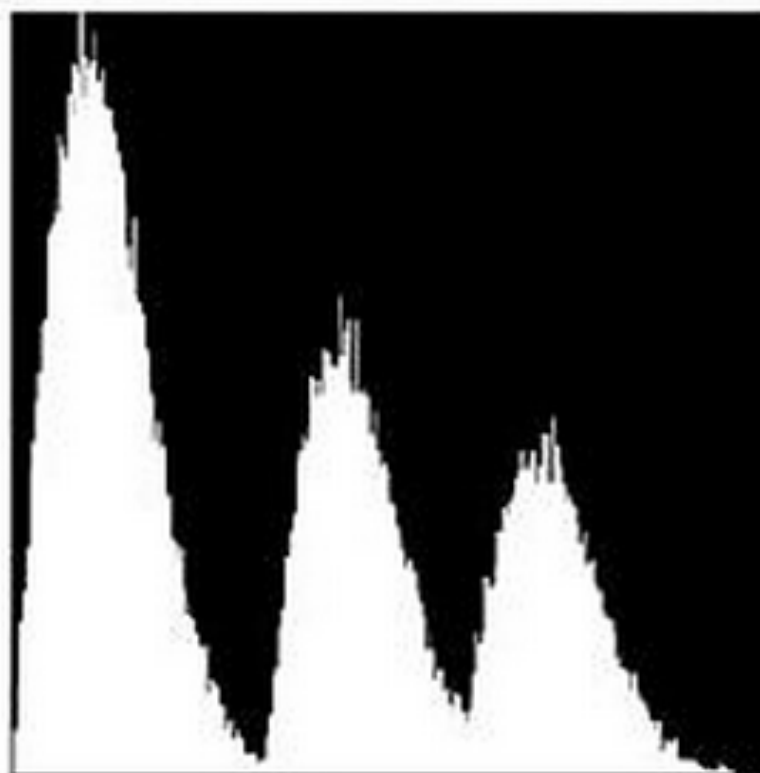


Image d'origine non bruitée





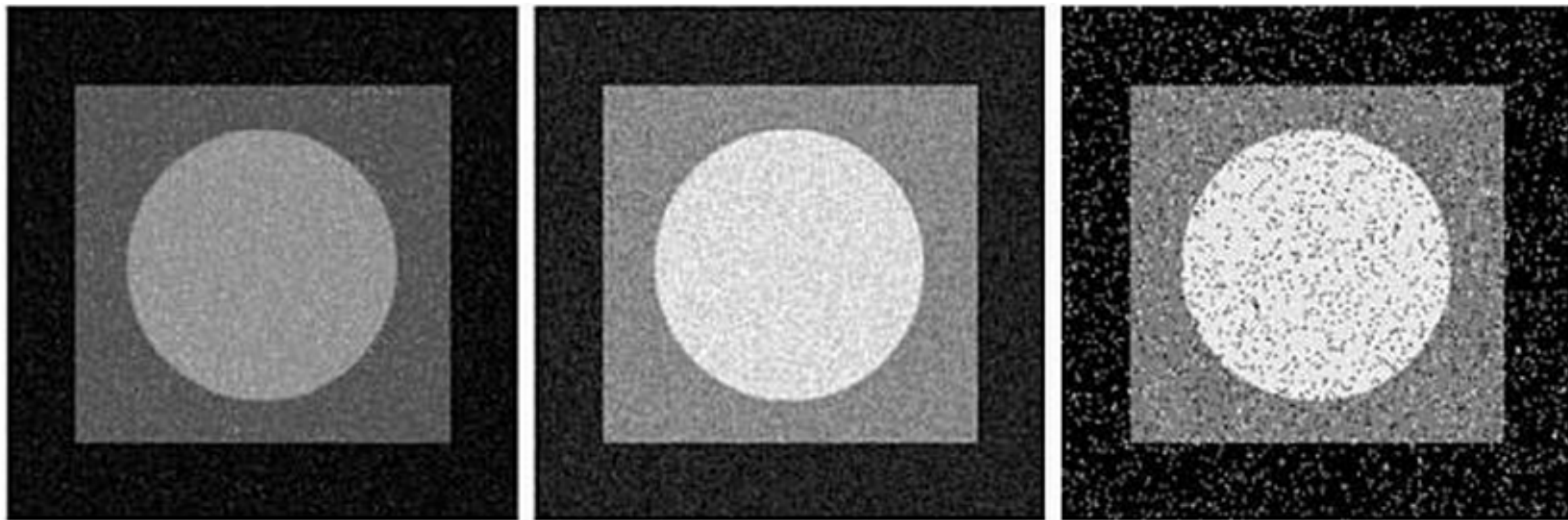
Gaussian

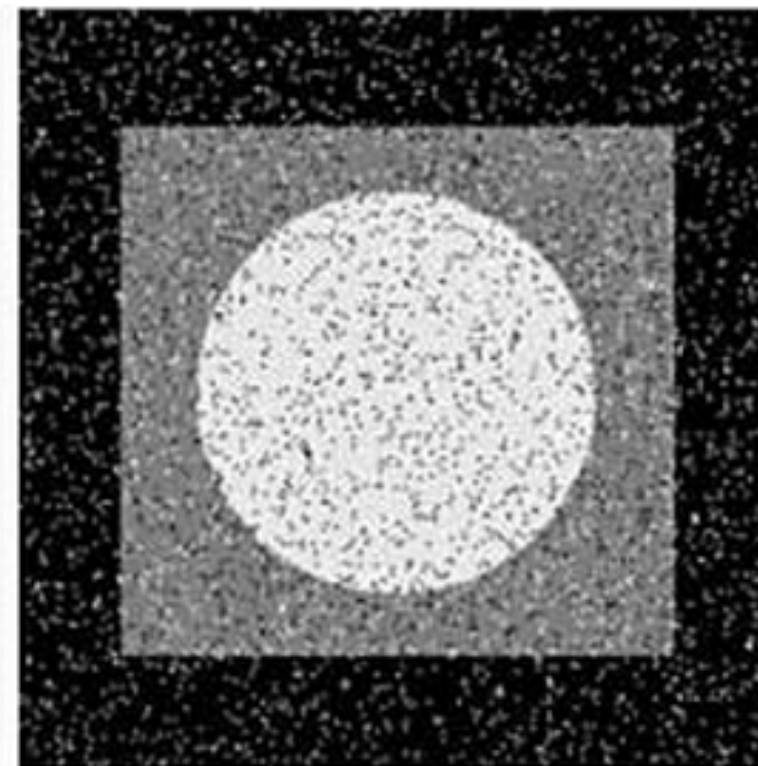
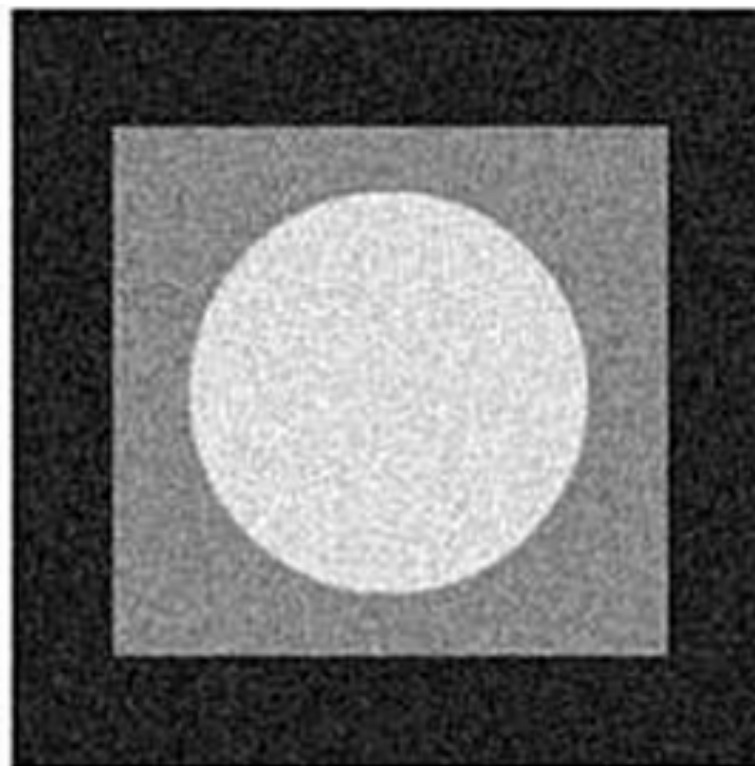
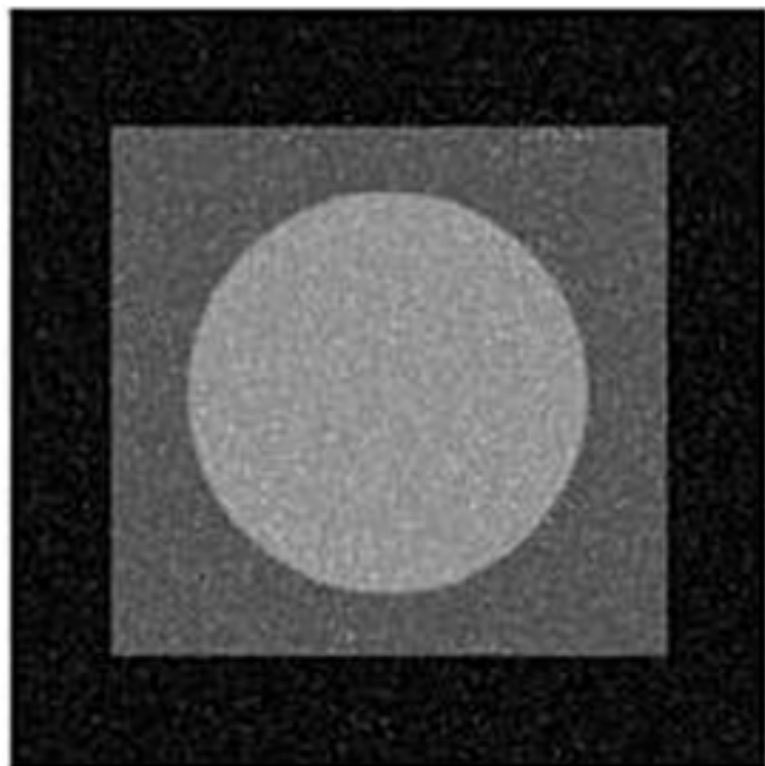


Rayleigh

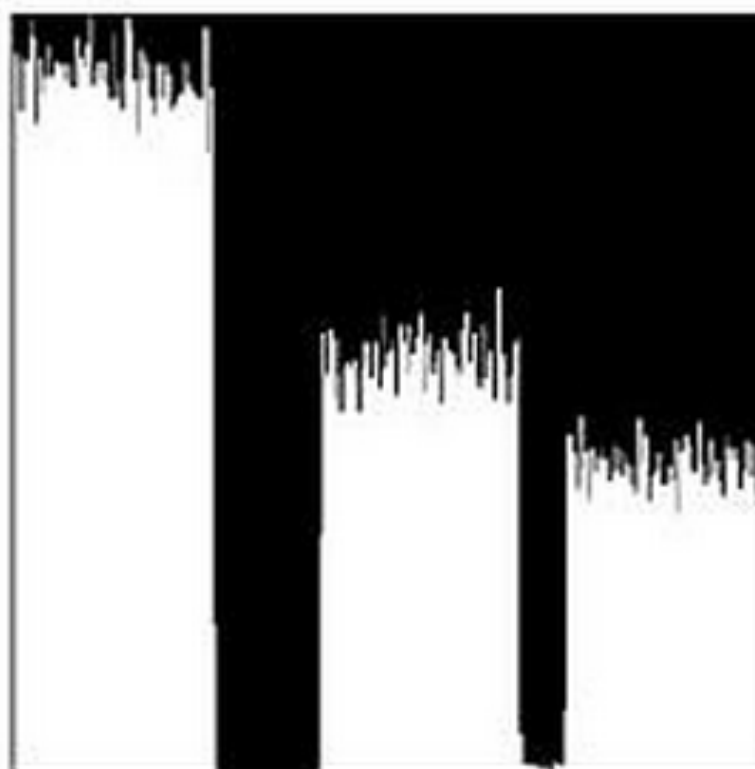


Gamma

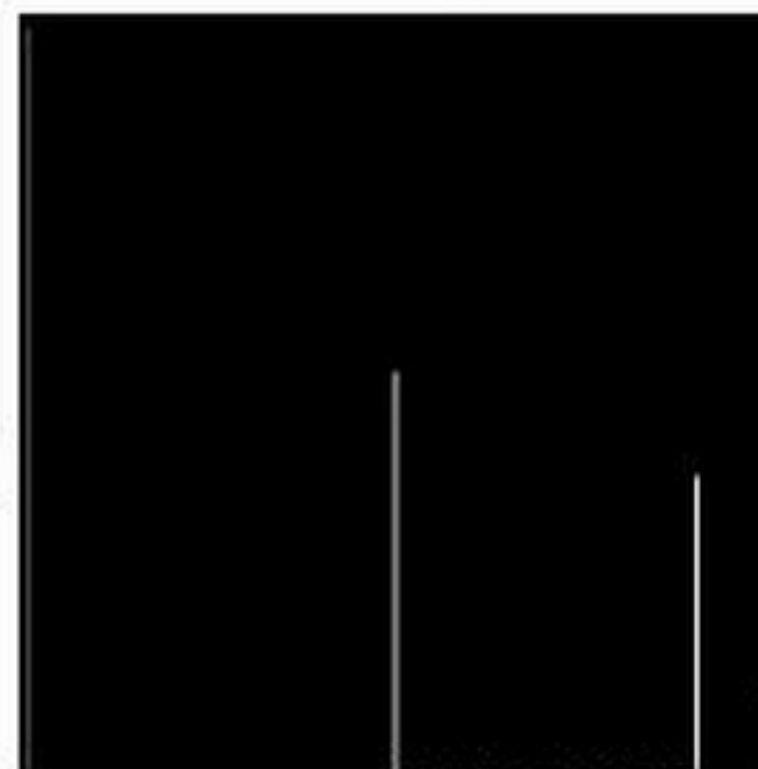




Exponential



Uniform



Salt & Pepper

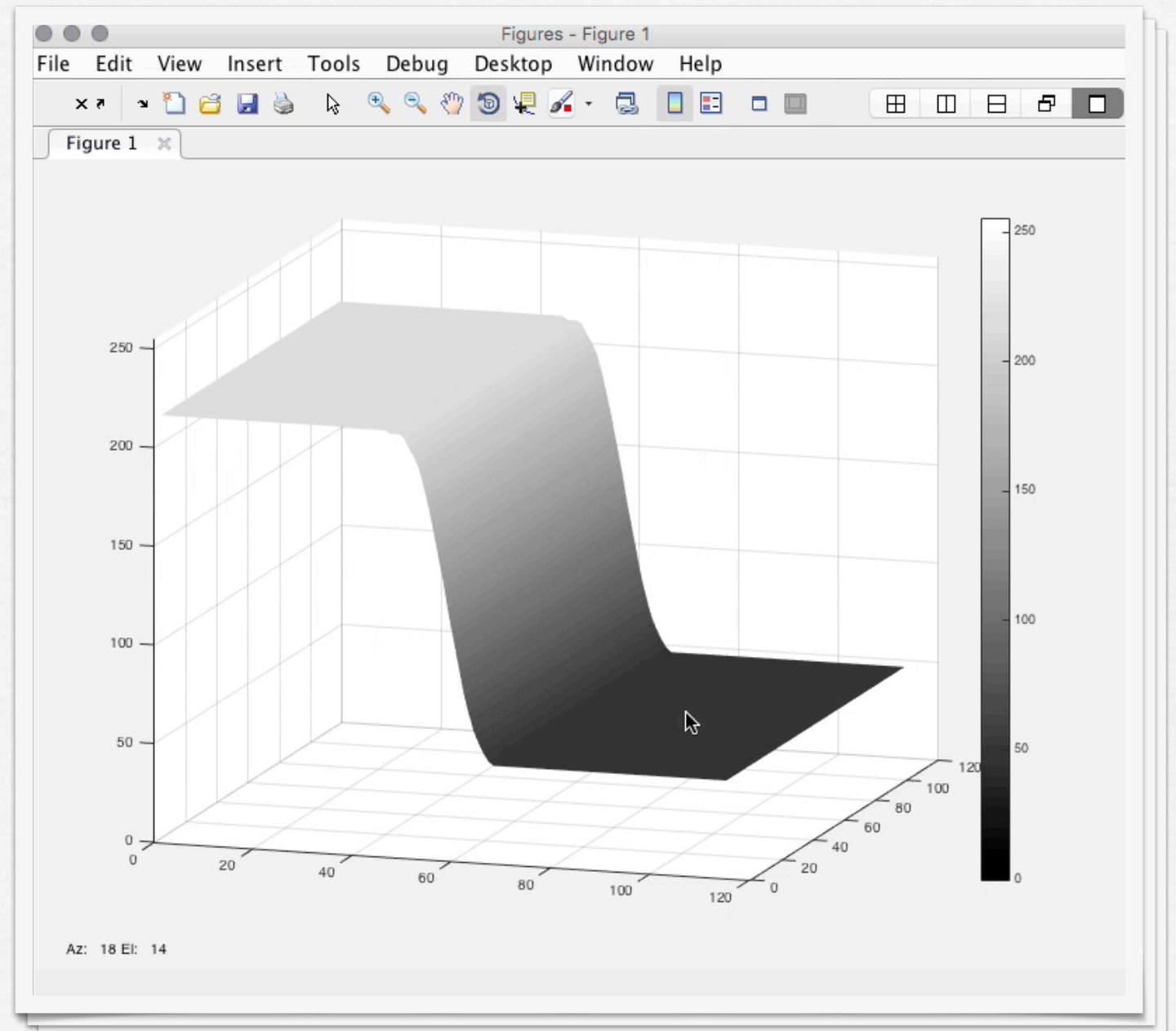
BRUIT, CONTRASTE, CONTOUR DANS UNE IMAGE

* Une image



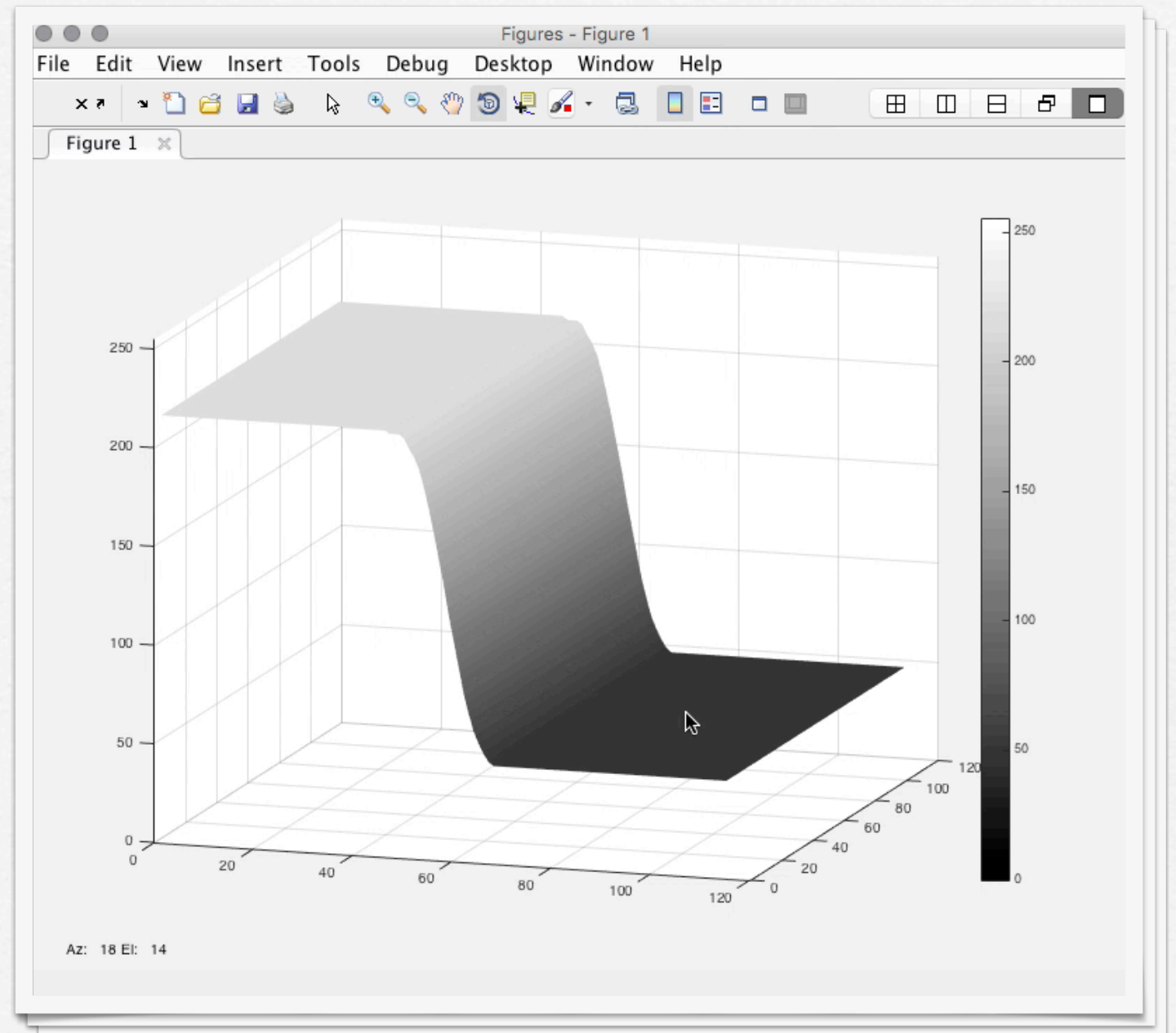
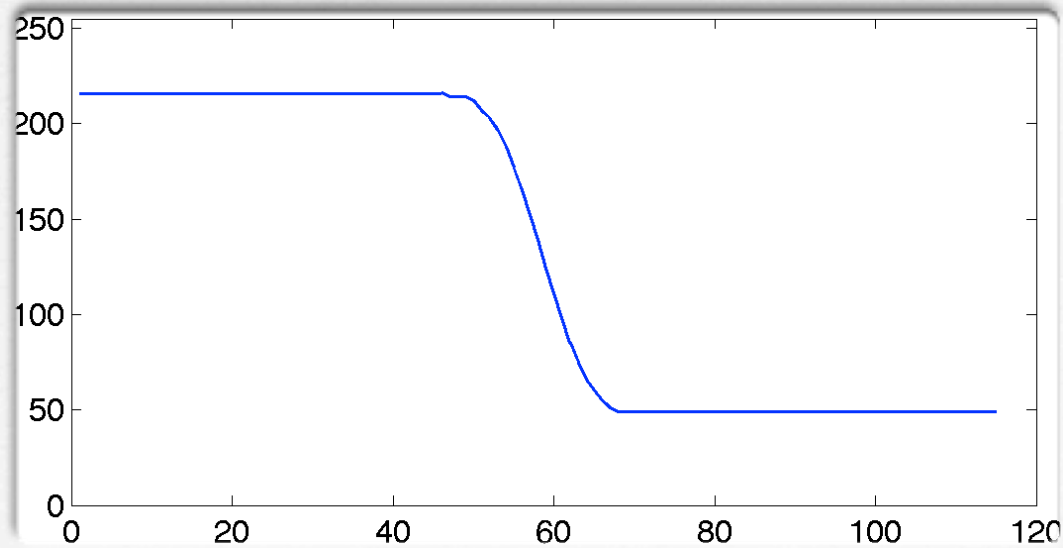
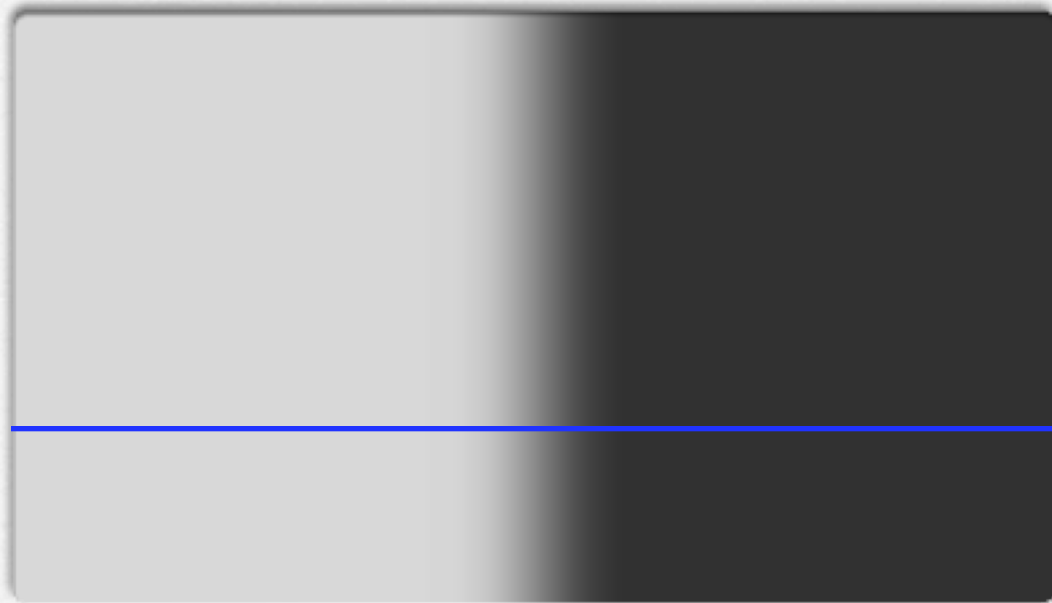
BRUIT, CONTRASTE, CONTOUR DANS UNE IMAGE

* Une image



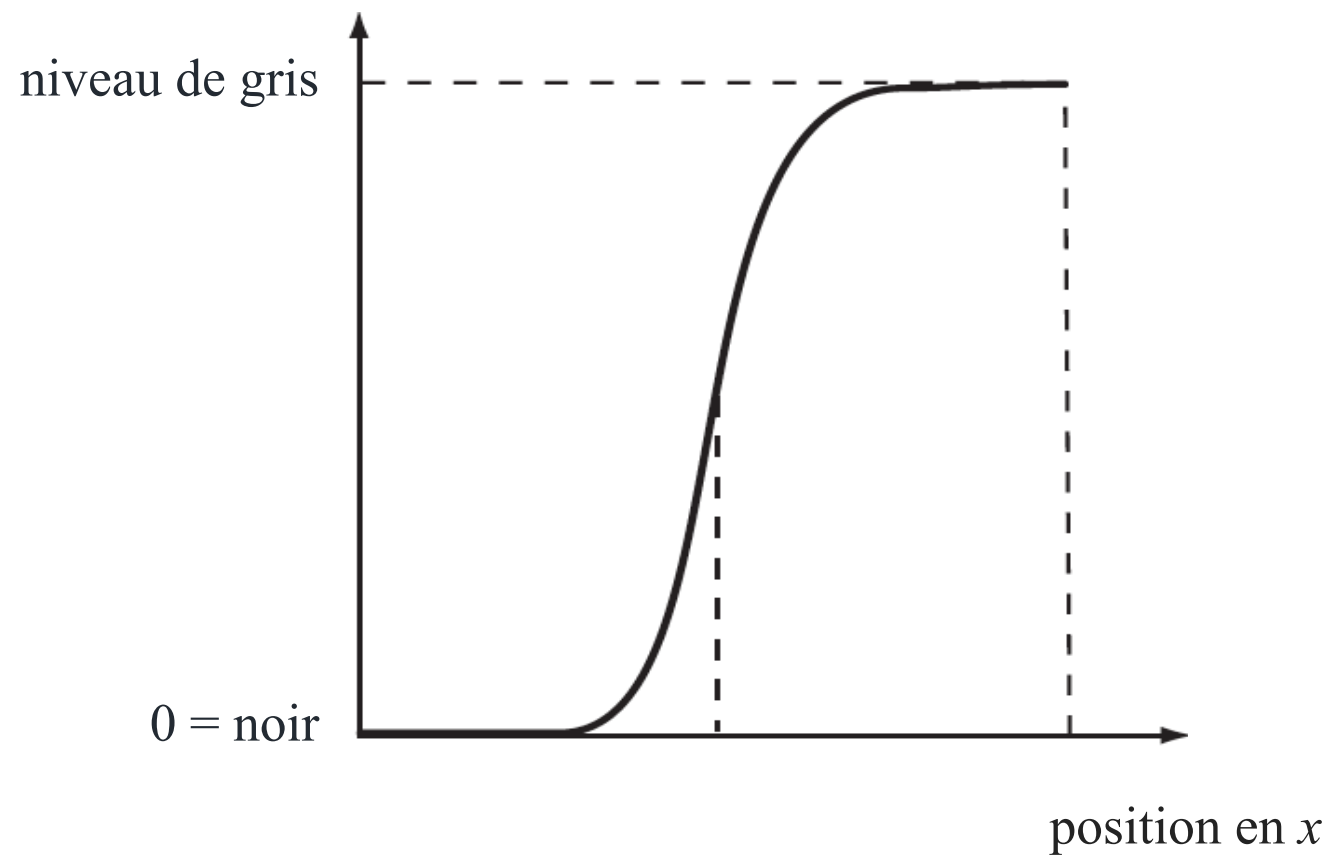
BRUIT, CONTRASTE, CONTOUR DANS UNE IMAGE

* Une image



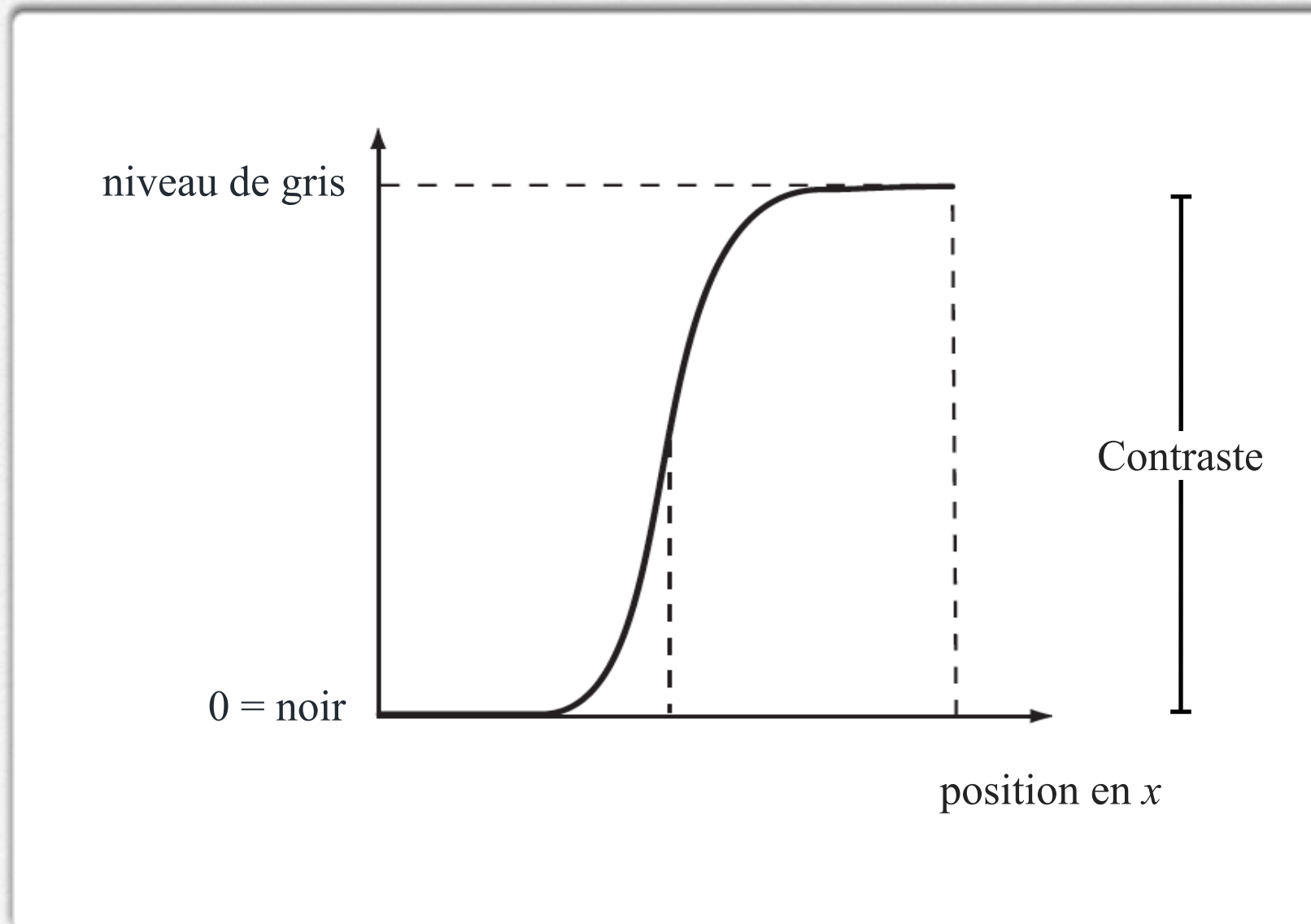
BRUIT, CONTRASTE, CONTOUR DANS UNE IMAGE

- * Distinguer le contraste de la netteté



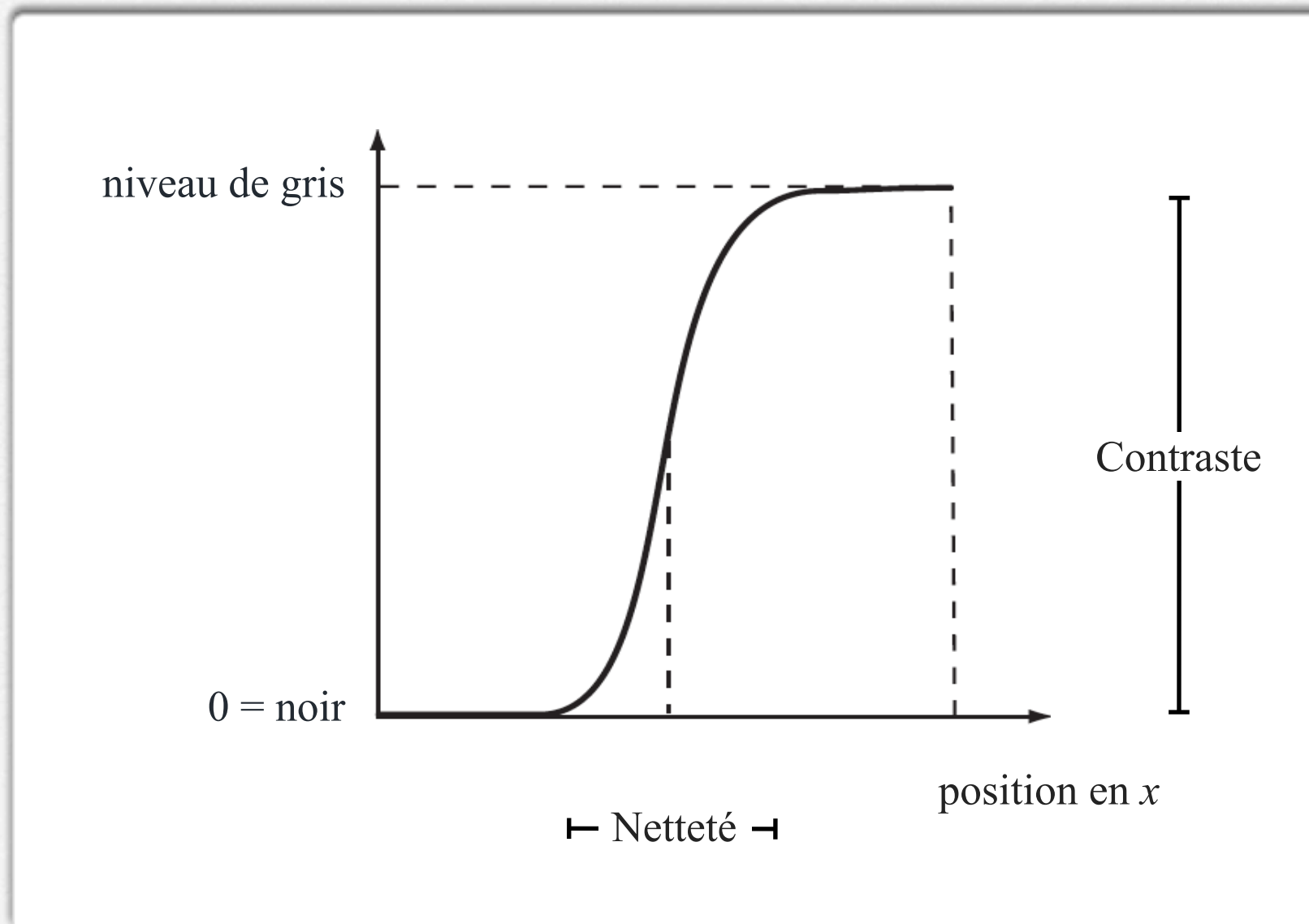
BRUIT, CONTRASTE, CONTOUR DANS UNE IMAGE

- * Distinguer le contraste de la netteté



BRUIT, CONTRASTE, CONTOUR DANS UNE IMAGE

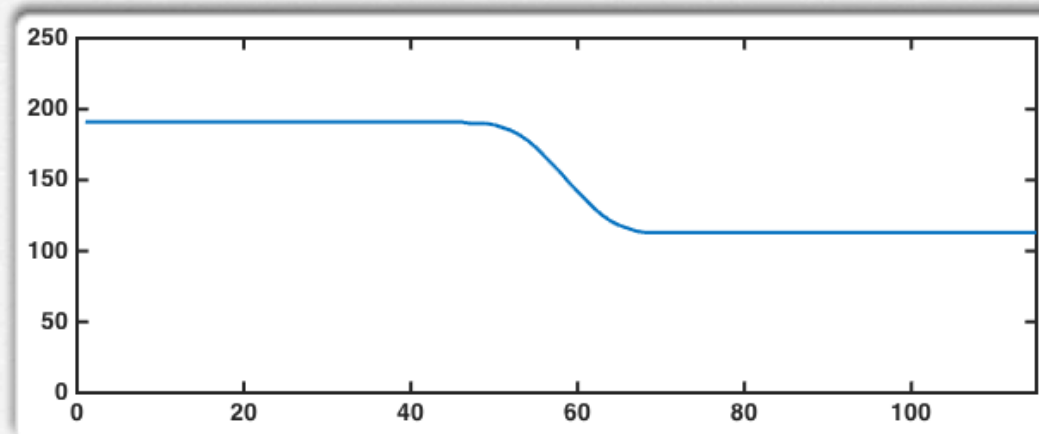
- * Distinguer le contraste de la netteté



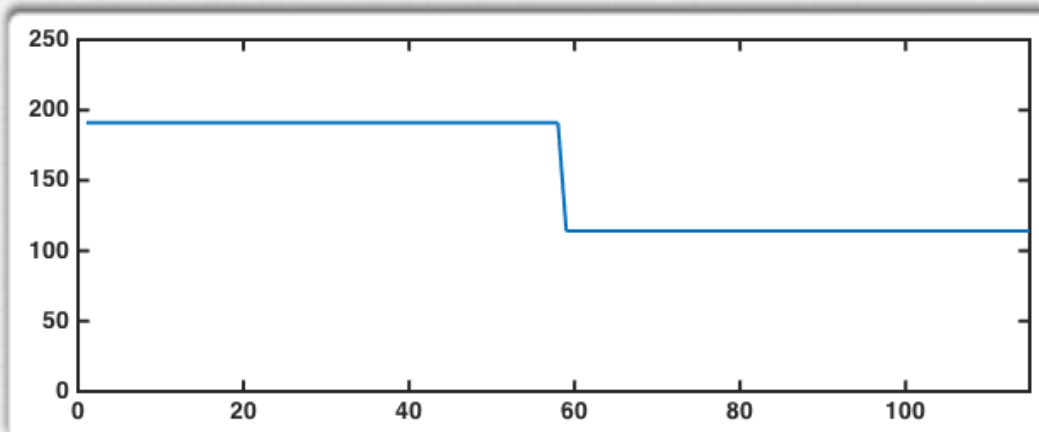
BRUIT, CONTRASTE, CONTOUR DANS UNE IMAGE

* Distinguer le contraste de la netteté

✓ Si un contour est flou, c'est parce que sa zone de transition entre les niveaux de gris est grande



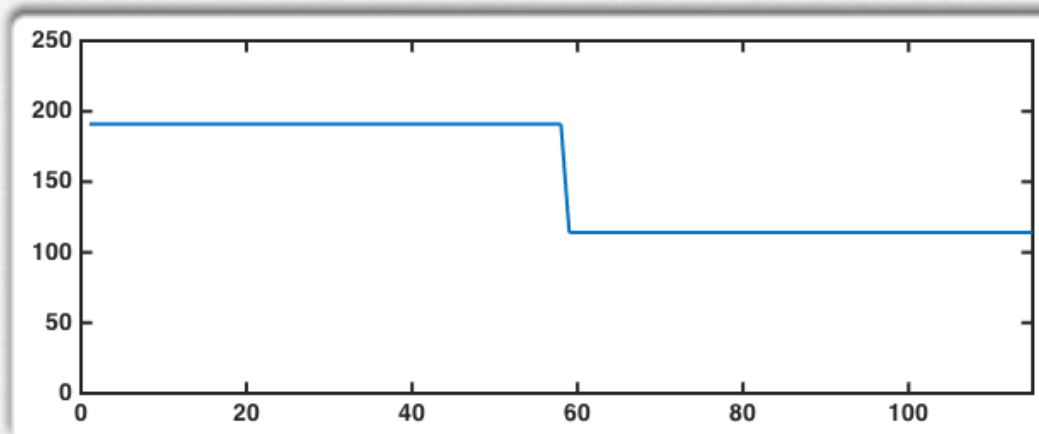
✓ On augmente la netteté



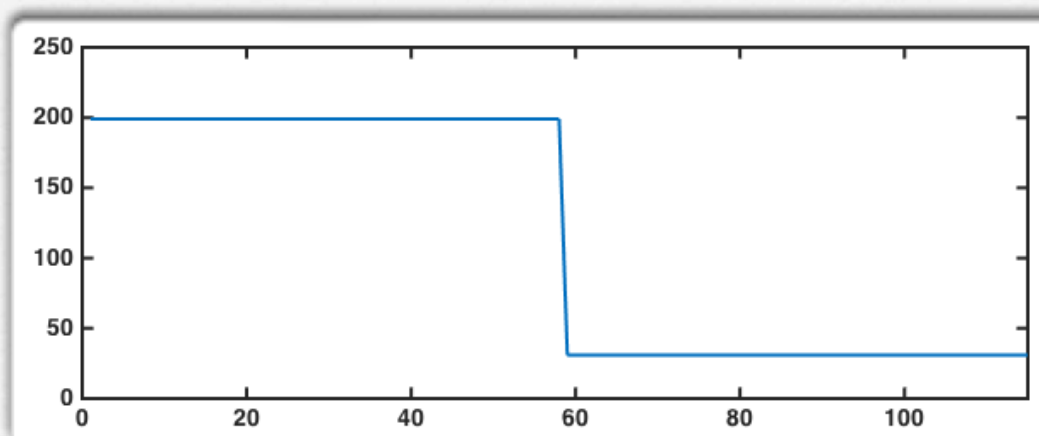
BRUIT, CONTRASTE, CONTOUR DANS UNE IMAGE

* Distinguer le contraste de la netteté

✓ Si un contour est peu contrasté, c'est parce les niveaux de gris d'un côté et de l'autre du contour sont proches



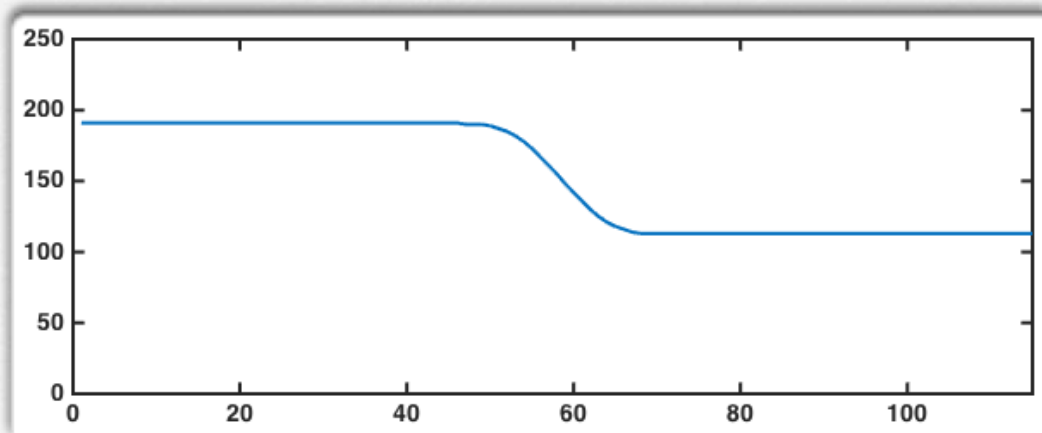
✓ On augmente le contraste



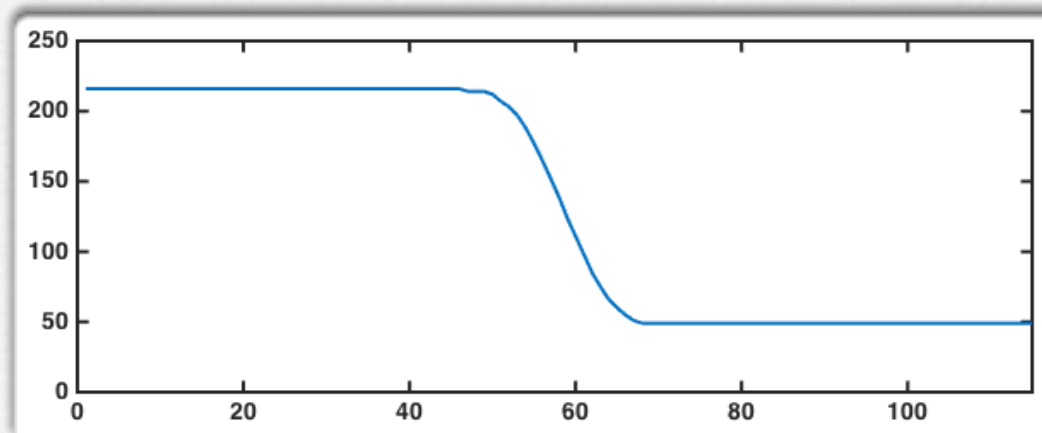
BRUIT, CONTRASTE, CONTOUR DANS UNE IMAGE

* Distinguer le contraste de la netteté

✓ Si un contour est peu contrasté, c'est parce les niveaux de gris d'un côté et de l'autre du contour sont proches



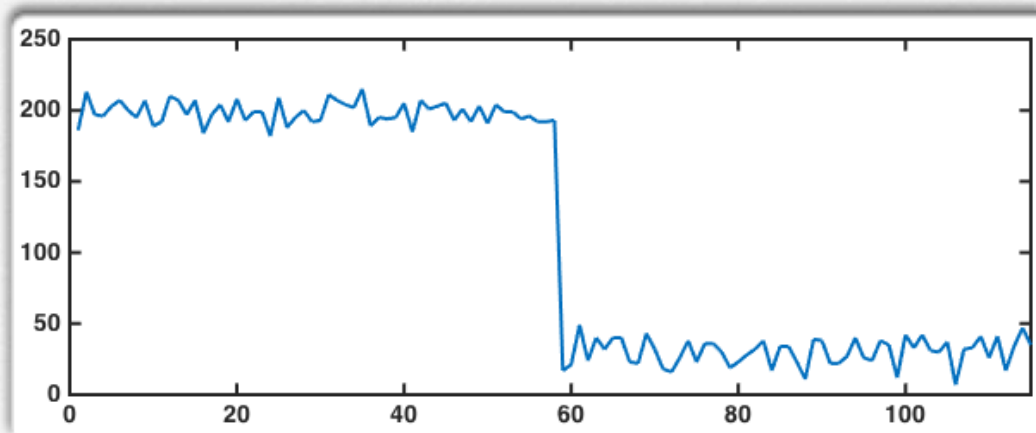
✓ Augmenter le contraste améliore l'impression de netteté



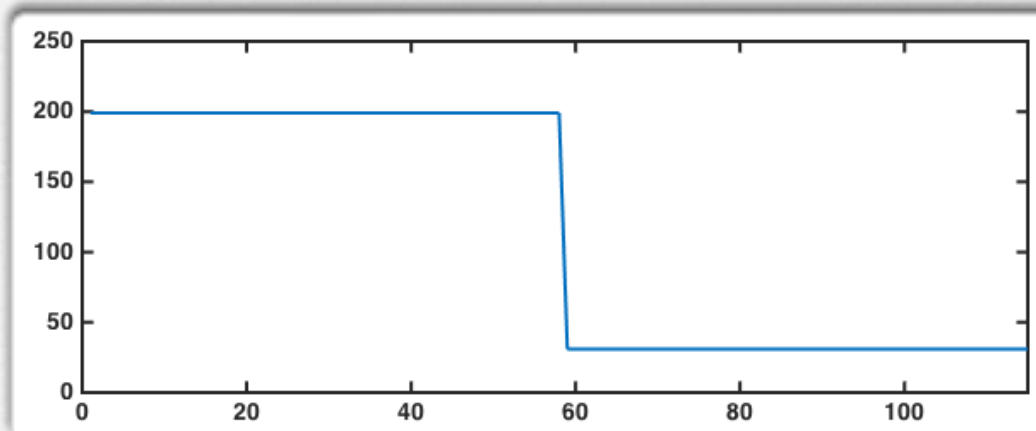
BRUIT, CONTRASTE, CONTOUR DANS UNE IMAGE

* Distinguer le bruit

✓ Si un contour est bruité, c'est parce des variations de n.g. surviennent dans les régions constantes



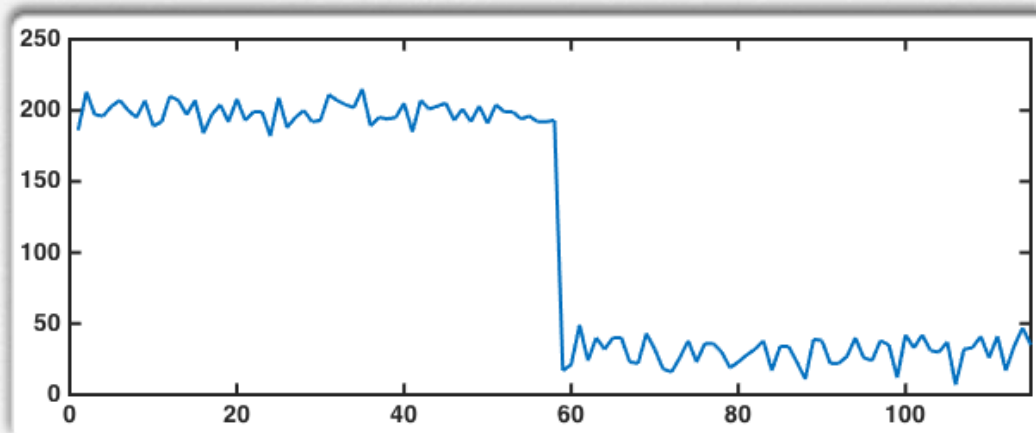
✓ Contour idéal : net, contrasté, non-bruité



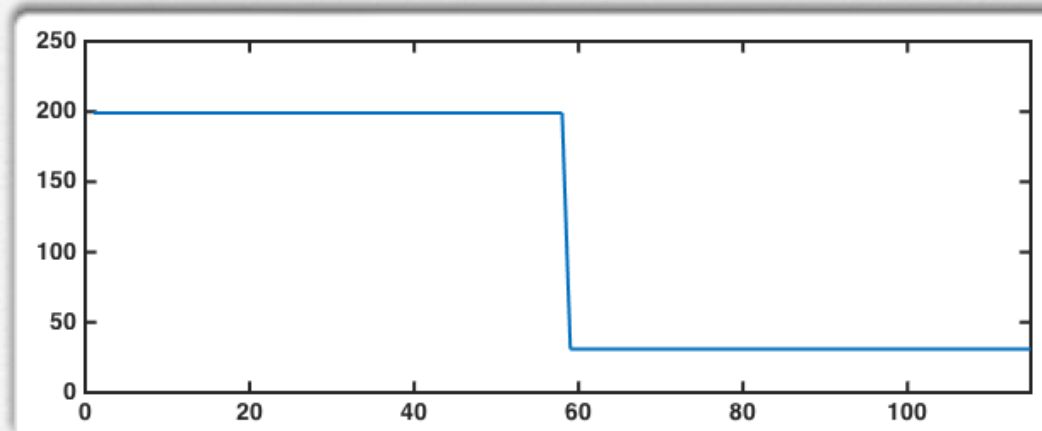
BRUIT, CONTRASTE, CONTOUR DANS UNE IMAGE

* Distinguer le bruit

- ✓ Si un contour est bruité, c'est parce des variations de n.g. surviennent dans les régions constantes



- ✓ Contour idéal : net, contrasté, non-bruité

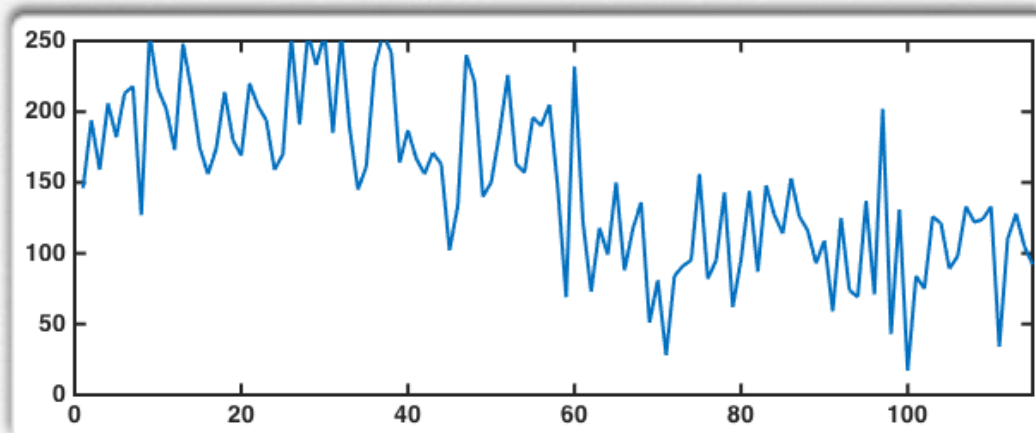


N'existe pas dans la vraie vie

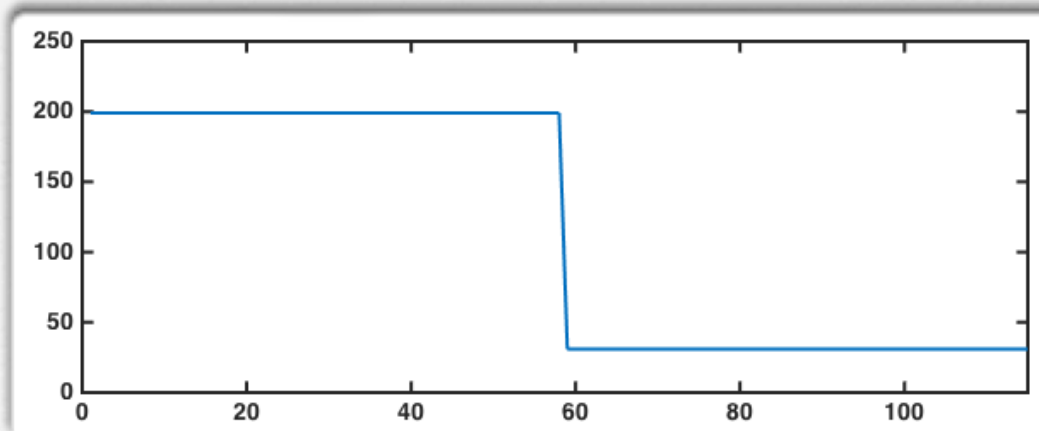
BRUIT, CONTRASTE, CONTOUR DANS UNE IMAGE

* Distinguer le bruit

✓ Un contour très bruité, peu contrasté et flou est difficile à localiser



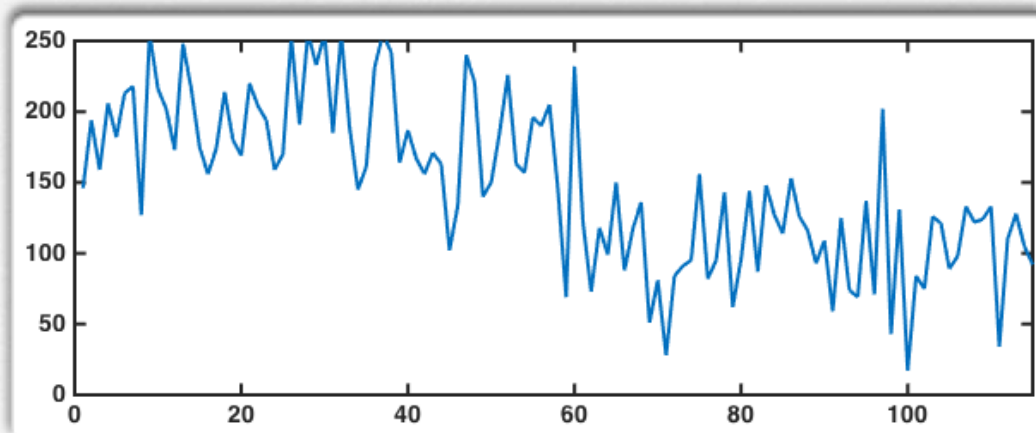
✓ Contour idéal : net, contrasté, non-bruité



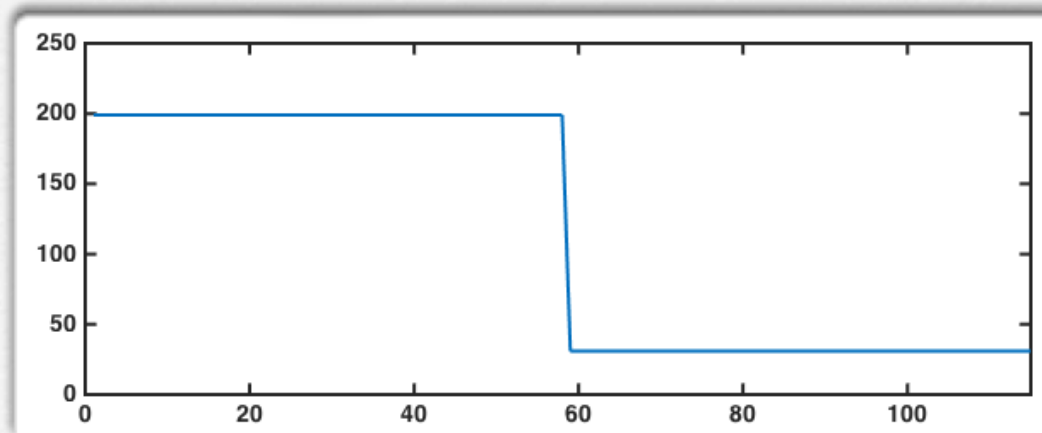
BRUIT, CONTRASTE, CONTOUR DANS UNE IMAGE

- * Distinguer le bruit

- ✓ Un contour très bruité, peu contrasté et flou est difficile à localiser



- ✓ Contour idéal : net, contrasté, non-bruité



N'existe pas dans la vraie vie



Métriques de qualité

Évaluation subjective de la qualité

L'objectif ici est de créer une mesure de qualité qui sera aussi fiable qu'un être humain. Pour ce faire, on doit d'abord s'intéresser à **l'évaluation subjective de la qualité** c'est-à-dire comprendre de façon empirique comment un être humain juge de la qualité d'une image.

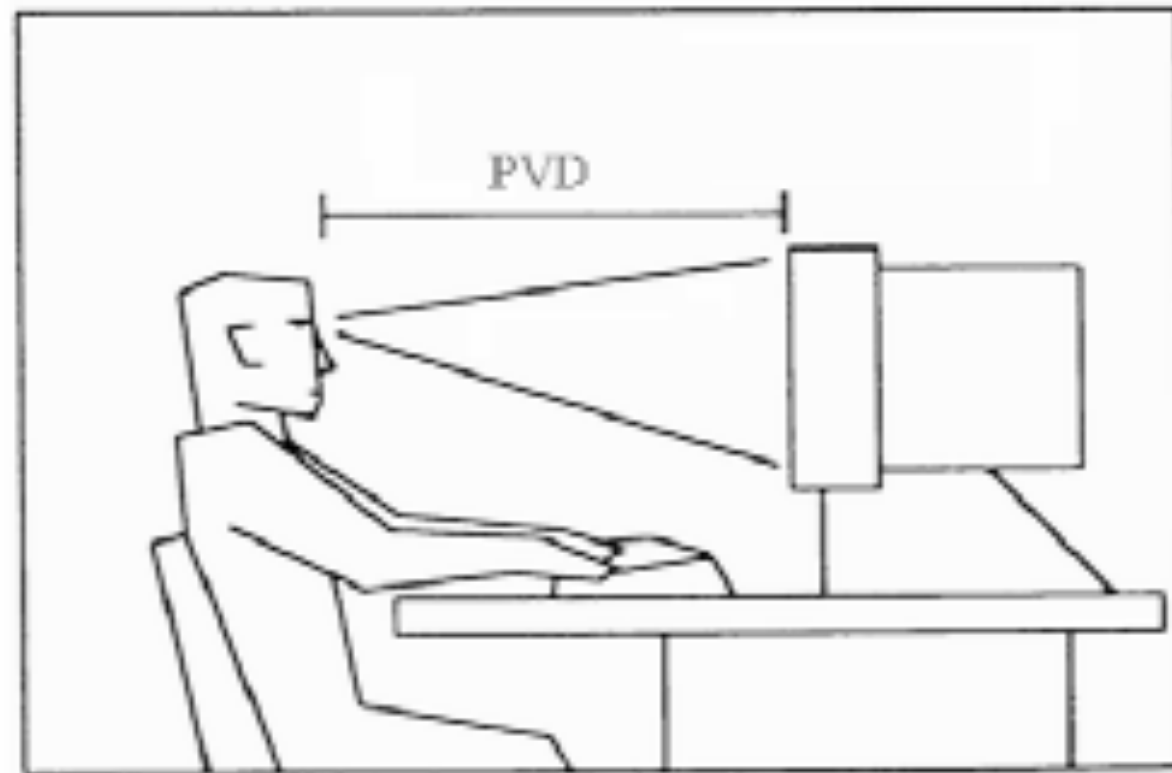


Figure 1 – *Conditions d'évaluation de la qualité*

Évaluation subjective de la qualité

L'objectif ici est de créer une mesure de qualité qui sera aussi fiable qu'un être humain. Pour ce faire, on doit d'abord s'intéresser à **l'évaluation subjective de la qualité** c'est-à-dire comprendre de façon empirique comment un être humain juge de la qualité d'une image.

Environnement normalisé

Standards ISO 3664 et ITU-R 500-10

- Distance d'environ 1 mètre à l'écran
- Angle d'observation d'environ 30 degrés
- Écran de haute qualité de 22 à 26 pouces
- Écran calibré (white balance, correction gamma, etc.)
- Chromaticité de l'arrière plan = illuminant D65 (D65 -> centre du diagramme de chromaticité)
- Sources de lumière D65
- Aucune lumière dans le champ visuel de l'observateur
- Aucune lumière devant refléter l'écran

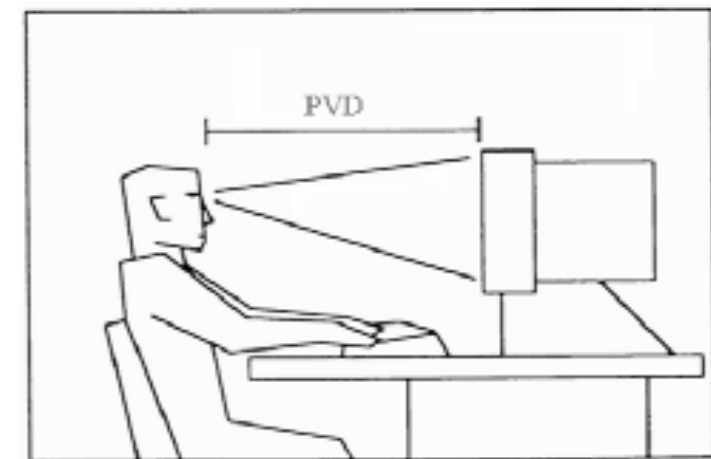


Figure 1 – Conditions d'évaluation de la qualité



Figure 2 – Exemple de salle d'évaluation

Évaluation subjective de la qualité

Séance d'évaluation

- Séance d'au plus 30 minutes
- Le score des 15-20 premières images doit être ignoré car tout observateur « stabilise » son opinion au début de la séance
- Au moins 15 observateurs.
- Les observateurs doivent avoir un appareil visuel normal

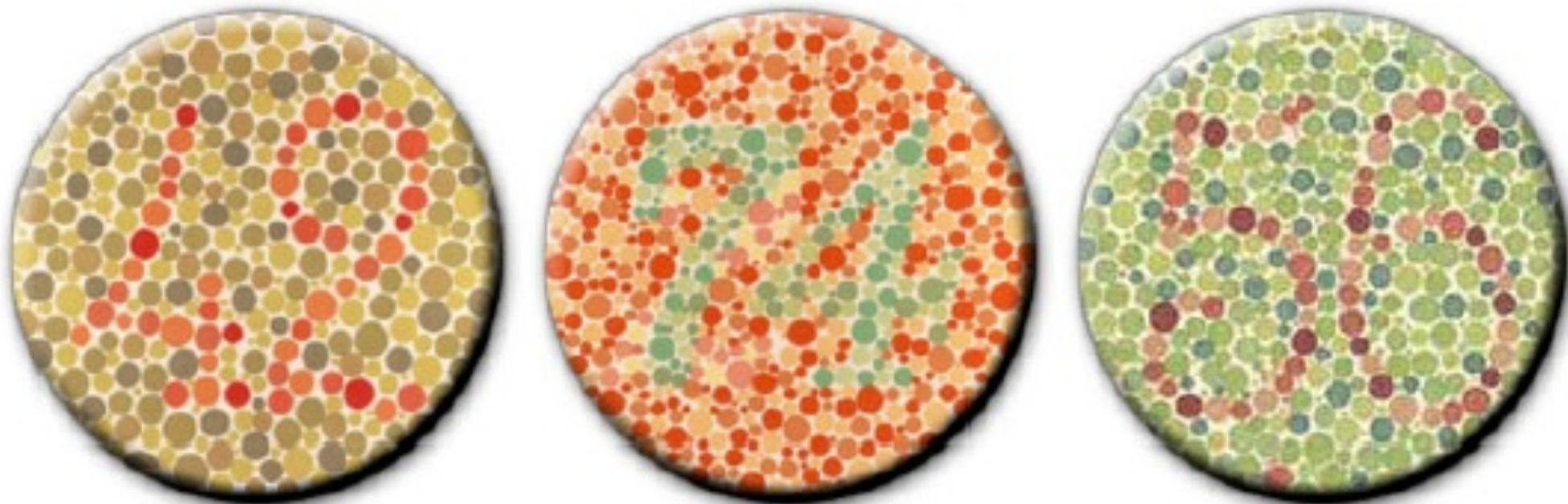


Figure 4: *Test d'Ishihara.*

Évaluation subjective de la qualité

Tests

On peut faire des **tests comparatifs** et des tests **de mesure absolue**. Les tests comparatifs sont généralement plus souvent retenus

Évaluation subjective de la qualité

Tests

On peut faire des **tests comparatifs** et des tests **de mesure absolue**. Les tests comparatifs sont généralement plus souvent retenus

Tests comparatifs : ordonner les images de la meilleure à la moins bonne



Artéfacts de compression

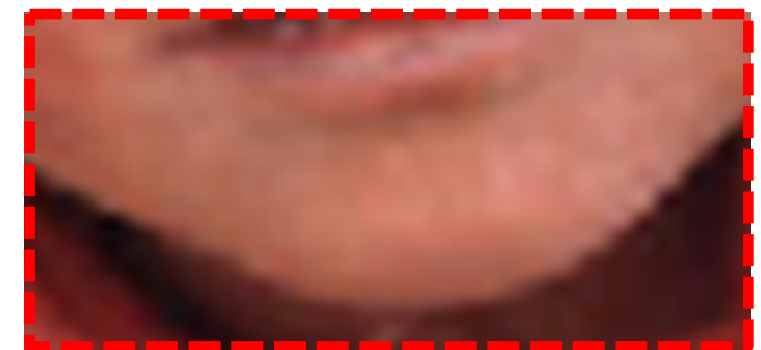
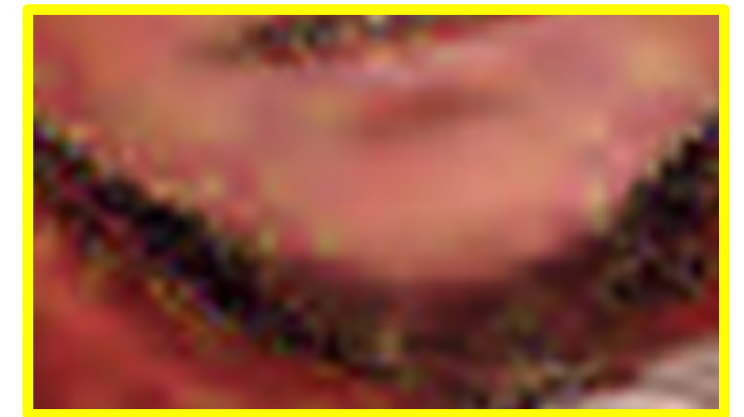
Figure 5 : *Test d'ordonnancement.*

Évaluation subjective de la qualité

Tests

On peut faire des **tests comparatifs** et des tests **de mesure absolue**. Les tests comparatifs sont généralement plus souvent retenus

Tests comparatifs : ordonner les images de la meilleure à la moins bonne



Artéfacts de compression

Figure 5 : *Test d'ordonnancement.*

Évaluation subjective de la qualité

Tests

On peut faire des **tests comparatifs** et des tests **de mesure absolue**. Les tests comparatifs sont généralement plus souvent retenus

Tests 2 à 2: Comparer l'image d'origine à sa version dégradée



VS



Artéfacts de compression



VS



Bruit

Évaluation subjective de la qualité

Tests

On peut faire des **tests comparatifs** et des tests **de mesure absolue**. Les tests comparatifs sont généralement plus souvent retenus

Tests 2 à 2: Comparer l'image d'origine à sa version dégradée

Echelle à cinq notes		
Qualité		Dégradation
Excellente	5	Imperceptible
Bonne	4	Perceptible mais non gênant
Assez bonne	3	Légèrement gênant
Médiocre	2	Gênant
Mauvaise	1	Très gênant

u_{ijk}

Évaluation subjective de la qualité

Tests

On peut faire des **tests comparatifs** et des tests **de mesure absolue**. Les tests comparatifs sont généralement plus souvent retenus

Tests 2 à 2: Comparer l'image d'origine à sa version dégradée

Echelle à cinq notes		
Qualité		Dégradation
Excellente	5	Imperceptible
Bonne	4	Perceptible mais non gênant
Assez bonne	3	Légèrement gênant
Médiocre	2	Gênant
Mauvaise	1	Très gênant

MOS (*mean opinion score*)

$$\bar{u}_{jk} = \frac{1}{N} \sum_{i=1}^N u_{ijk}$$

Où u_{ijk} est la note de l'observateur i pour la dégradation j de l'image k . N est le nombre d'observateurs

Évaluation subjective de la qualité

Tests

On peut faire des **tests comparatifs** et des tests **de mesure absolue**. Les tests comparatifs sont généralement plus souvent retenus

Test de mesure absolue : Donner une note de 1 à 5 à une image dégradée

Echelle à cinq notes		
Qualité		Dégradation
Excellente	5	Imperceptible
Bonne	4	Perceptible mais non gênant
Assez bonne	3	Légèrement gênant
Médiocre	2	Gênant
Mauvaise	1	Très gênant

MOS (*mean opinion score*)

$$\bar{u}_{jk} = \frac{1}{N} \sum_{i=1}^N u_{ijk}$$

Où u_{ijk} est la note de l'observateur i pour la dégradation j de l'image k . N est le nombre d'observateurs

Métriques de qualité

Le but est de trouver une métrique de qualité qui saura **imiter le MOS**. Les métriques de mesure absolue sont relativement rares, souvent complexe et vont au-delà du cadre de ce cours. Les métriques les plus fréquentes sont les **métriques d'évaluation 2 à 2**.

Erreur quadratique moyenne (*mean square error*)

$$MSE(f, g) = \frac{1}{nm} \sum_{i,j} (f(i, j) - g(i, j))^2$$

Métriques de qualité

Le but est de trouver une métrique de qualité qui saura **imiter le MOS**. Les métriques de mesure absolue sont relativement rares, souvent complexe et vont au-delà du cadre de ce cours. Les métriques les plus fréquentes sont les **métriques d'évaluation 2 à 2**.

Erreur quadratique moyenne (*mean square error*)

$$MSE(f, g) = \frac{1}{nm} \sum_{i,j} (f(i, j) - g(i, j))^2$$

Ratio signal sur bruit (*signal to noise ratio*)

$$\begin{aligned} SNR(f, g) &= 10 \log \frac{P_f}{P_{f-g}} \\ &= 10 \log \frac{\sum_{ij} f(i, j)^2}{\sum_{ij} (f(i, j) - g(i, j))^2} \\ &= 20 \log \frac{\sum_{ij} f(i, j)}{\sum_{ij} f(i, j) - g(i, j)} \end{aligned}$$

Métriques de qualité

Une métrique beaucoup plus souvent utilisée est le PSNR.

Ratio signal sur bruit impulsionnel (*Peak signal to noise ratio*)

$$PSNR(f, g) = 10 \log \left(\frac{d^2}{MSE(f, g)} \right)$$

où d est la valeur maximale du signal (ici 255)

Métriques de qualité

Une métrique de qualité est une fonction qui mesure la **qualité visuelle** d'une image. Ainsi plus une image est dégradée par du flou, du bruit ou des artefacts de compression (e.g. JPEG) plus la qualité sera faible.



Qualité faible



Qualité bonne



Qualité excellente

Métriques de qualité

Les métriques de qualité sont souvent utilisées pour évaluer les méthodes de débruitage et de déconvolution

Exemple :



Image d'origine f



Image dégradée g



Image débruitée par la méthode 1



Image débruitée par la méthode 2

Par conséquent la méthode 1
est meilleure que la méthode 2

MSSIM sera défini dans les diapos
plus loin

Métriques de qualité

Les métriques de qualité sont souvent utilisées pour évaluer les méthodes de débruitage et de déconvolution

Exemple :



Image d'origine f



Image dégradée g



f_1

Image débruitée par la méthode 1



Image débruitée par la méthode 2

Par conséquent la méthode 1
est meilleure que la méthode 2

MSSIM sera défini dans les diapos
plus loin

Métriques de qualité

Les métriques de qualité sont souvent utilisées pour évaluer les méthodes de débruitage et de déconvolution

Exemple :



Image d'origine f



Image dégradée g



f_1

Image débruitée par la méthode 1



f_2

Image débruitée par la méthode 2

Par conséquent la méthode 1
est meilleure que la méthode 2

MSSIM sera défini dans les diapos
plus loin

Métriques de qualité

Les métriques de qualité sont souvent utilisées pour évaluer les méthodes de débruitage et de déconvolution

Exemple :



Image d'origine f



Image dégradée g



f_1

Image débruitée par la méthode 1



f_2

Image débruitée par la méthode 2

$$MSSIM(f, f_1) > MSSIM(f, f_2)$$

Par conséquent la méthode 1
est meilleure que la méthode 2

MSSIM sera défini dans les diapos
plus loin

Métriques de qualité

Le problème avec MSE, SNR et PSNR est qu'une légère modification d'une image, parfois même imperceptible par l'œil humain, peut avoir un effet majeur sur ces métriques. Par conséquent, d'autres métriques ont été proposés. Parmi les plus utilisées sont

UQI (Universal Quality Index)
SSIM (Structural SIMilarity)

Z Wang, A Bovik, **A Universal Image Quality Index**, IEEE Signal Processing Letters, 13(4), 2002

Z Wang, A Bovik, H. Sheikh, E. Simoncelli **Image Quality Assessment: From Error Visibility to Structural Similarity**, IEEE TIP, 13(4), 2004

Métriques de qualité

UQI:

μ_f, μ_g : moyenne des images f et g

σ_f, σ_g : écart – type des images f et g

σ_{fg} : covariance $\frac{1}{nm} \sum_{i,j} (f_{ij} - \mu_f)(g_{ij} - \mu_g)$

$$UQI(f, g) = \frac{4\sigma_{fg}\mu_f\mu_g}{(\sigma_f^2 + \sigma_g^2)(\mu_f^2 + \mu_g^2)}$$

Métriques de qualité

UQI:

Les auteurs précisent qu'il est préférable d'utiliser une version locale de UQI:

$$UQI(f, g) = \sum_i UQI(f_i, g_i)$$

où

f_j, g_j : est le contenu de l'image à l'intérieur de la j -ème petite fenêtre (par exemple 11x11)

UQI meilleure que MSE

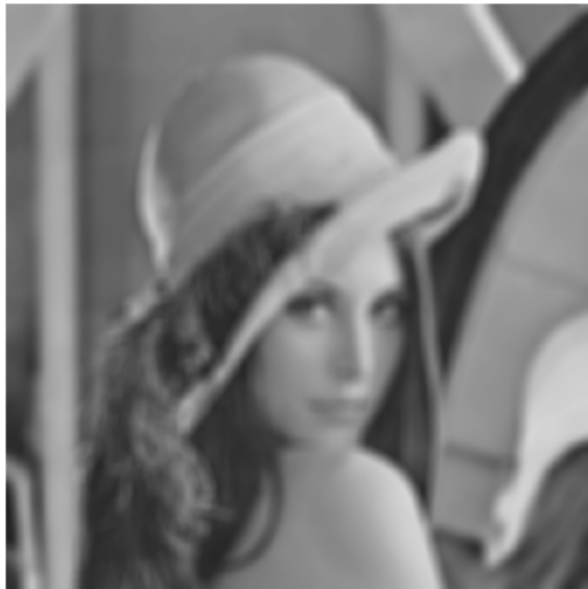
(Images dégradées ayant la même MSE)

Image originale

MSE : 225
UQI : 0.937

MSE : 225
UQI : 0.989

MSE : 225
UQI : 0.64



MSE : 225
UQI : 0.34

MSE : 225
UQI : 0.29

MSE : 225
UQI : 0.44

MSE : 225
UQI : 0.40

Métriques de qualité

SSIM:

Métriques de qualité

SSIM:

$$l(f, g) = \frac{2\mu_f \mu_g + C_1}{\mu_f^2 + \mu_g^2 + C_1} \quad c(f, g) = \frac{2\sigma_f \sigma_g + C_2}{\sigma_f^2 + \sigma_g^2 + C_2} \quad s(f, g) = \frac{2\sigma_{fg} + C_3}{\sigma_f \sigma_g + C_3}$$

μ_f, μ_g : moyenne des images f et g

σ_f, σ_g : écart – type des images f et g

σ_{fg} : covariance $\frac{1}{nm} \sum_{i,j} (f_{ij} - \mu_f)(g_{ij} - \mu_g)$

C_1, C_2, C_3 : constantes

$$SSIM(f, g) = l(f, g)^\alpha c(f, g)^\beta s(f, g)^\gamma$$

Métriques de qualité

Tout comme UQI, il est préférable d'utiliser une **version locale** de SSIM, une version qu'ils nomment *mean structural similarity* **MSSIM**

$$MSSIM(f, g) = \frac{1}{N} \sum_j SSIM(f_j, g_j)$$

où

f_j, g_j : est le contenu de l'image à l'intérieur de la j -ème petite fenêtre (par exemple 11x11)

Artéfacts de
bruit gaussien



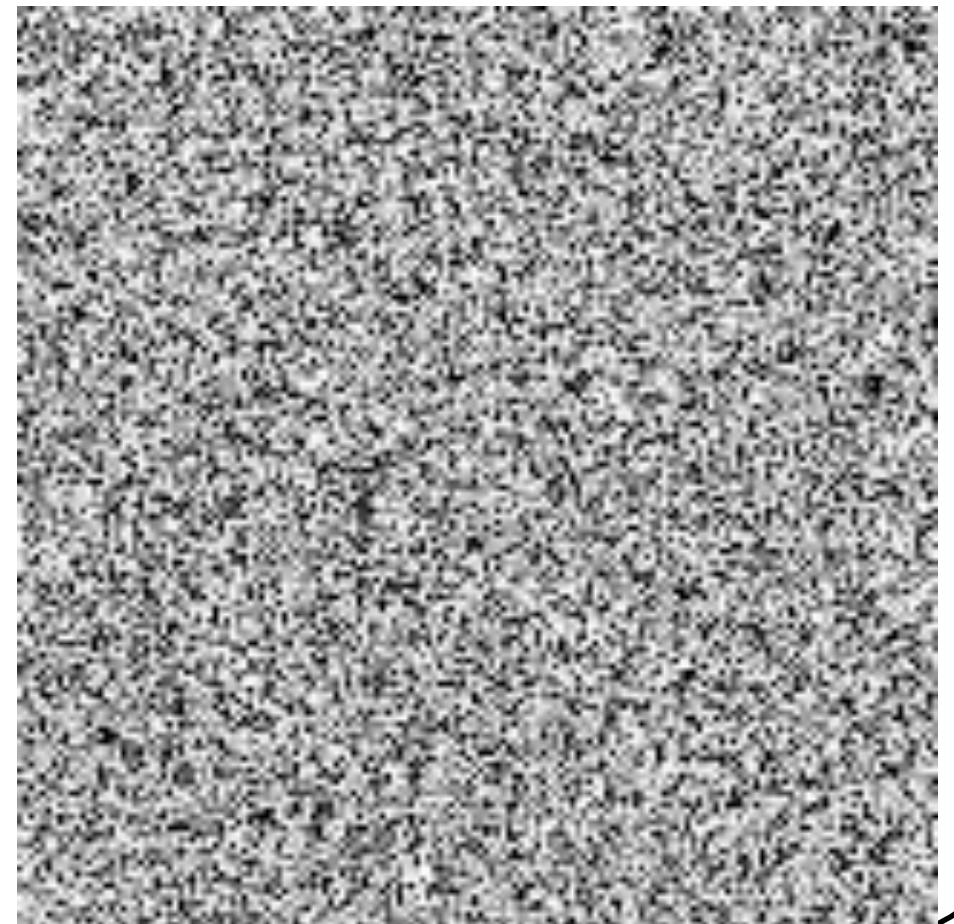
Image
originale



Valeur de SSIM
pour chaque
pixel



MSE



Artéfacts de
blocs
(compression
JPEG)



Image
originale



Valeur de SSIM
pour chaque
pixel



MSE



Artéfacts de
compression
(JPEG2000)



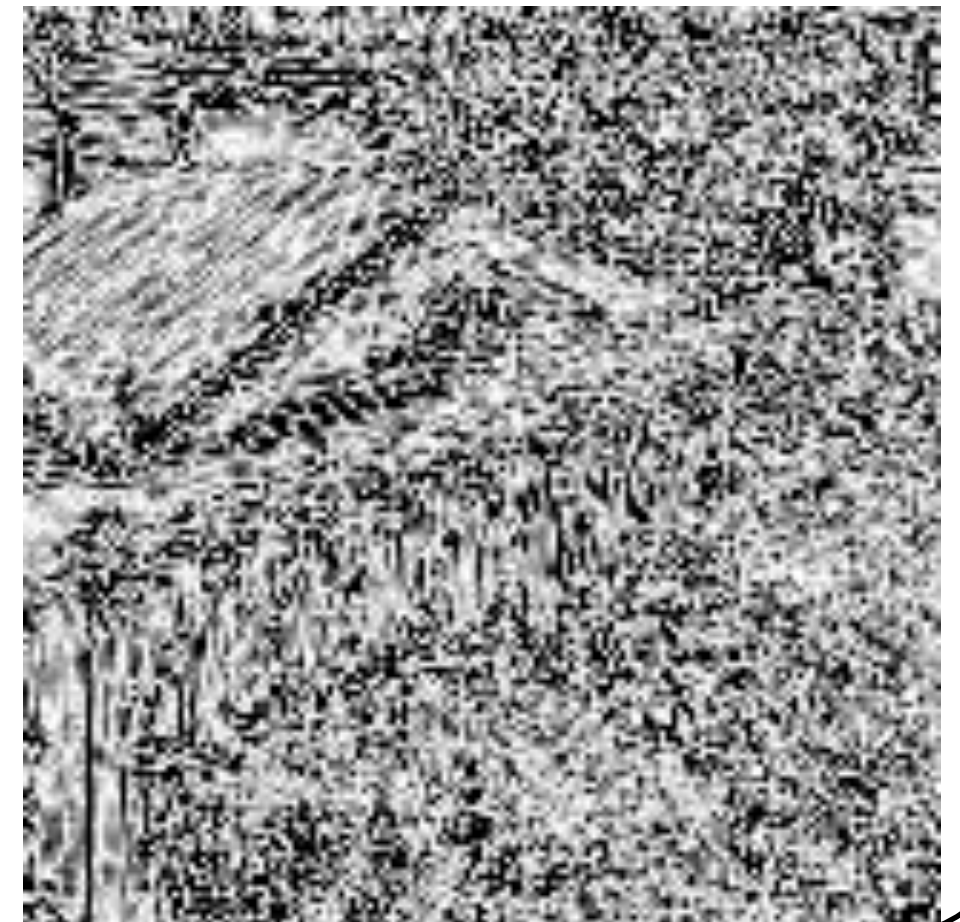
Image
originale



Valeur de SSIM
pour chaque
pixel



MSE



Autre exemple (images dégradées ayant la même MSE)



MSE=0, MSSIM=1



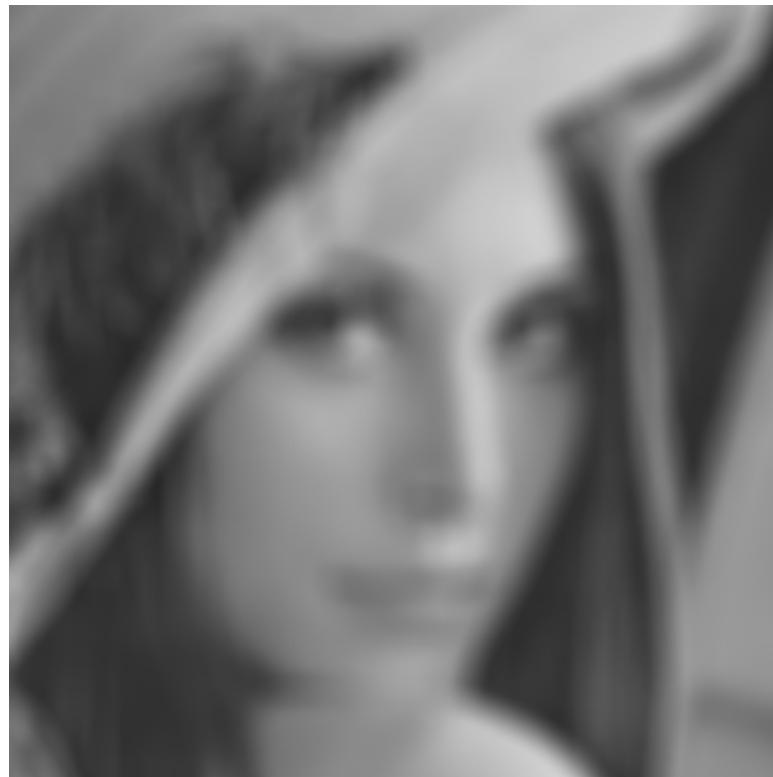
MSE=225, MSSIM=0.949



MSE=225, MSSIM=0.989



MSE=225, MSSIM=0.671



MSE=225, MSSIM=0.688



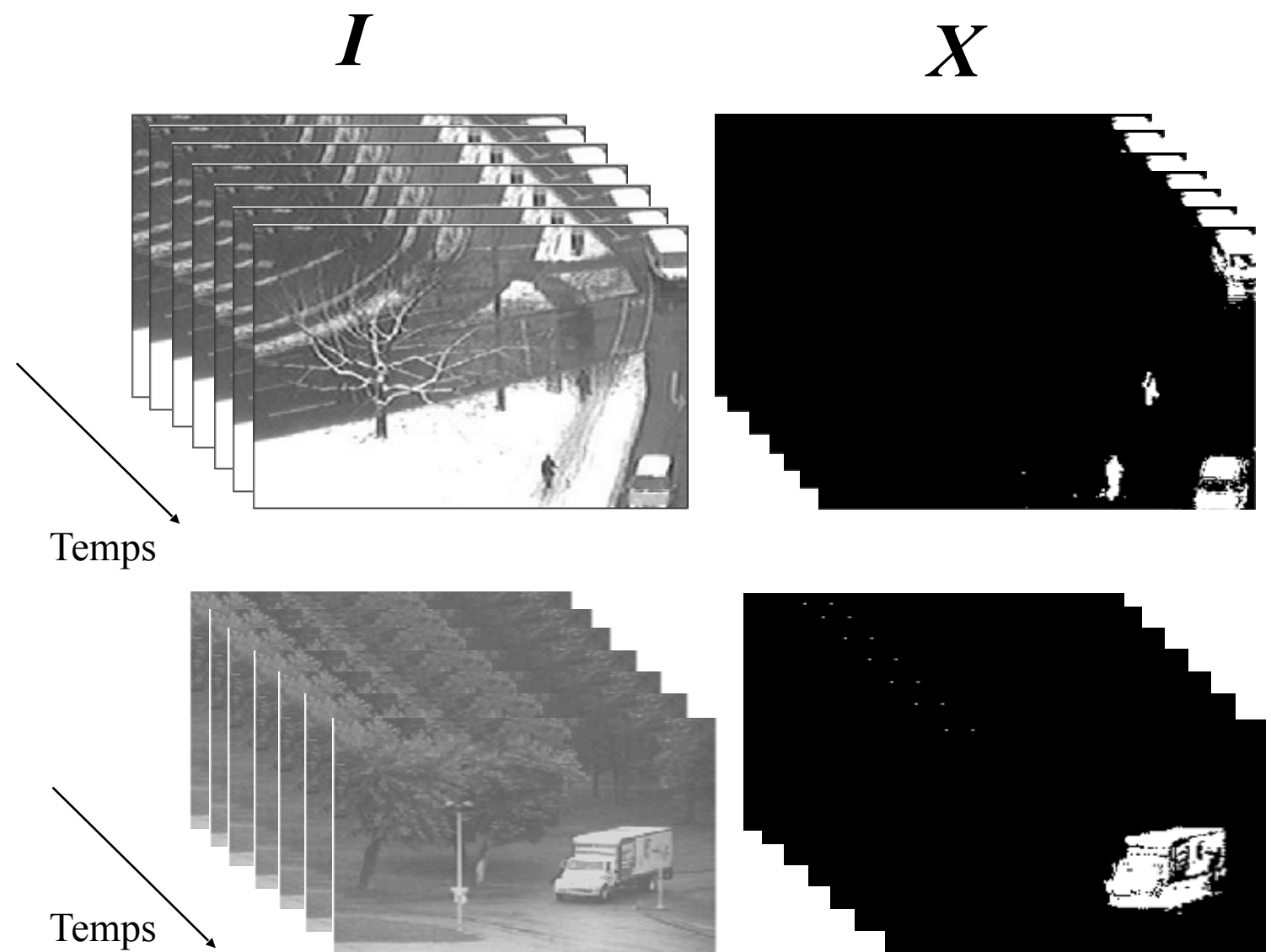
MSE=225, MSSIM=0.723

4. Opérations inter-images

Opérations inter-images.

Détection de mouvement

Le but? différencier (ou *segmenter*) les zones mobiles des zones immobiles dans une séquence vidéo.



Opérations inter-images.

Détection de mouvement

Le but? différencier (ou *segmenter*) les zones mobiles des zones immobiles dans une séquence vidéo.

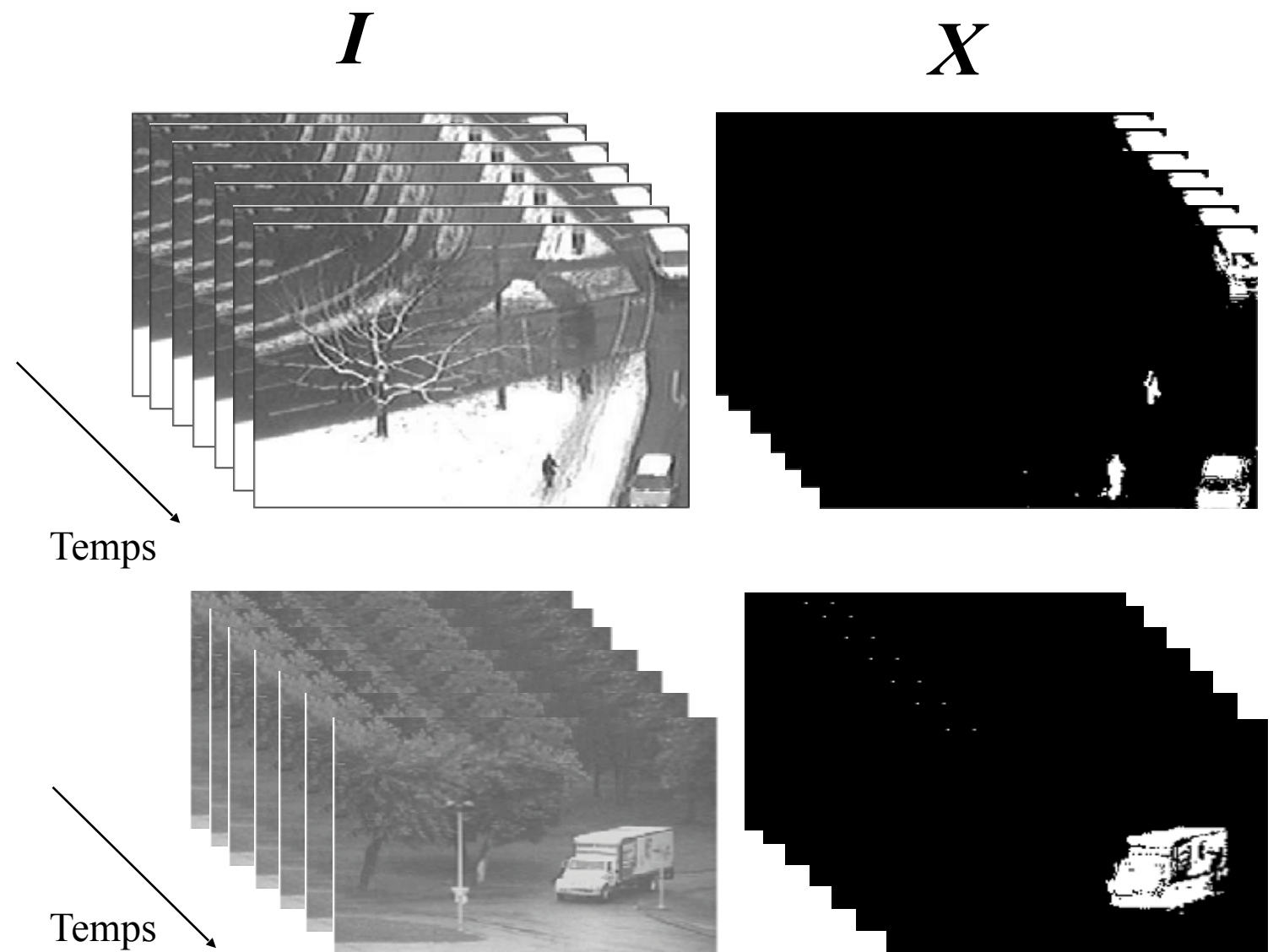
Ce dont nous disposons:

une séquence vidéo I constituée de N images, une à chaque temps t : I_t .

On appelle souvent « *frame* » une image dans une séquence vidéo.

Ce qu'il nous faut estimer:

un champ d'étiquettes X_t pour chaque frame I_t . En général, X_t est une image binaire de la même taille que I_t .



Opérations inter-images.

Détection de mouvement

Le but? différencier (ou *segmenter*) les zones mobiles des zones immobiles dans une séquence vidéo.

Deux grandes familles de méthodes :

(1) celles avec caméra **fixe**;

(2) celles avec caméra **mobile**. —————> Sujet de cours avancés

Les méthodes avec caméra **fixe** sont très utiles pour les applications de surveillance et de contrôle de la qualité (sur une ligne de montage par exemple). Pour ces méthodes, on considère souvent que:

$$I_t = B + \text{Objets en mouvement} \quad \text{où } B \text{ est un image du fond (background)}$$

On émet souvent deux hypothèses:

(1) les objets en mouvement ont une couleur différente des pixels du fond.

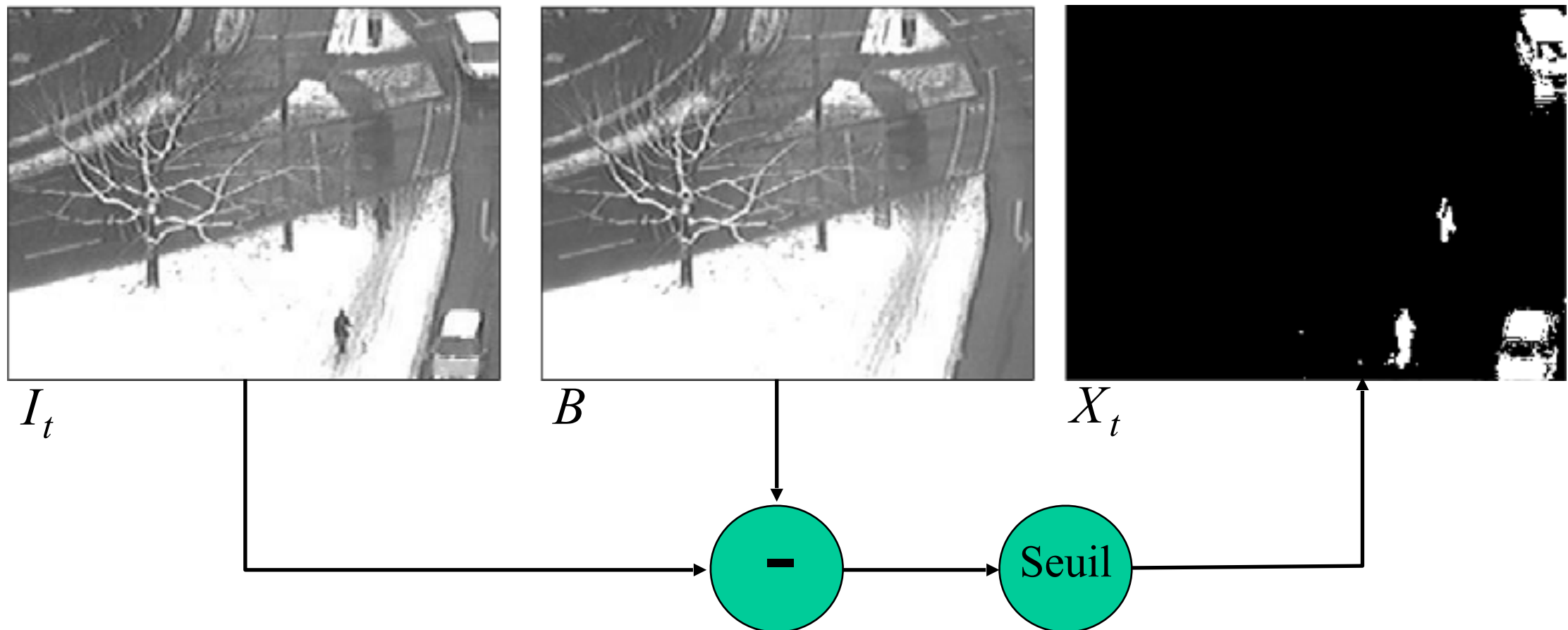
(2) l'image du fond est toujours la même, du début à la fin.

Opérations inter-images.

Détection de mouvement

En admettant pour vraies ces hypothèses, on peut estimer X par une simple opération de **soustraction de fond** :

$$X_t(x, y) = \text{Seuil}(|I_t(x, y) - B(x, y)|)$$

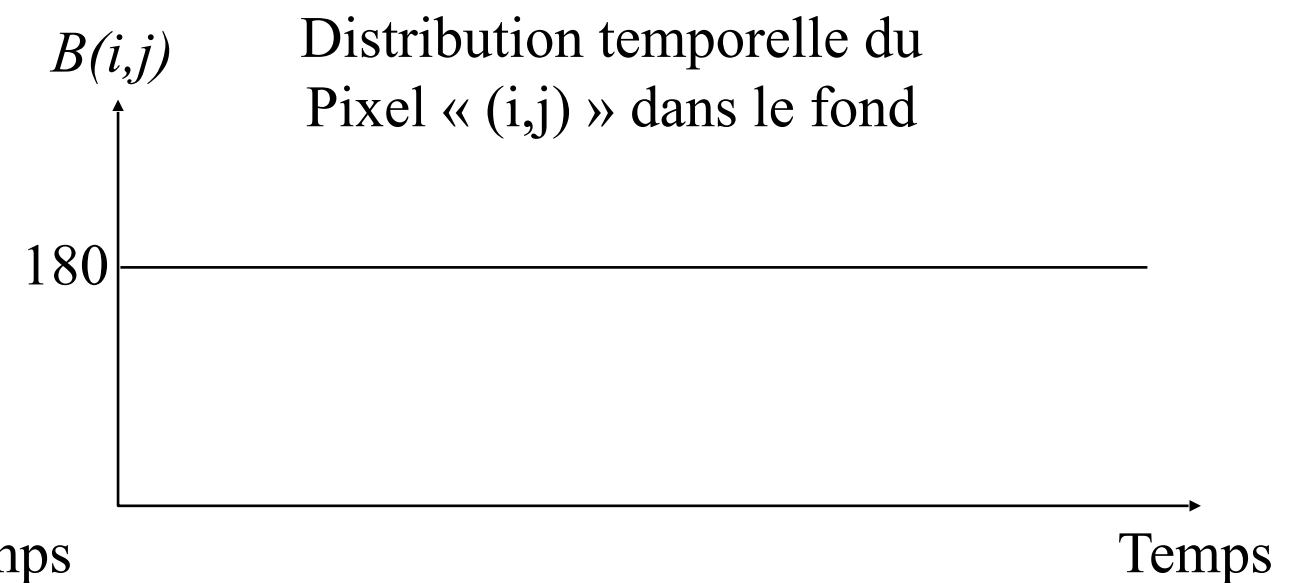
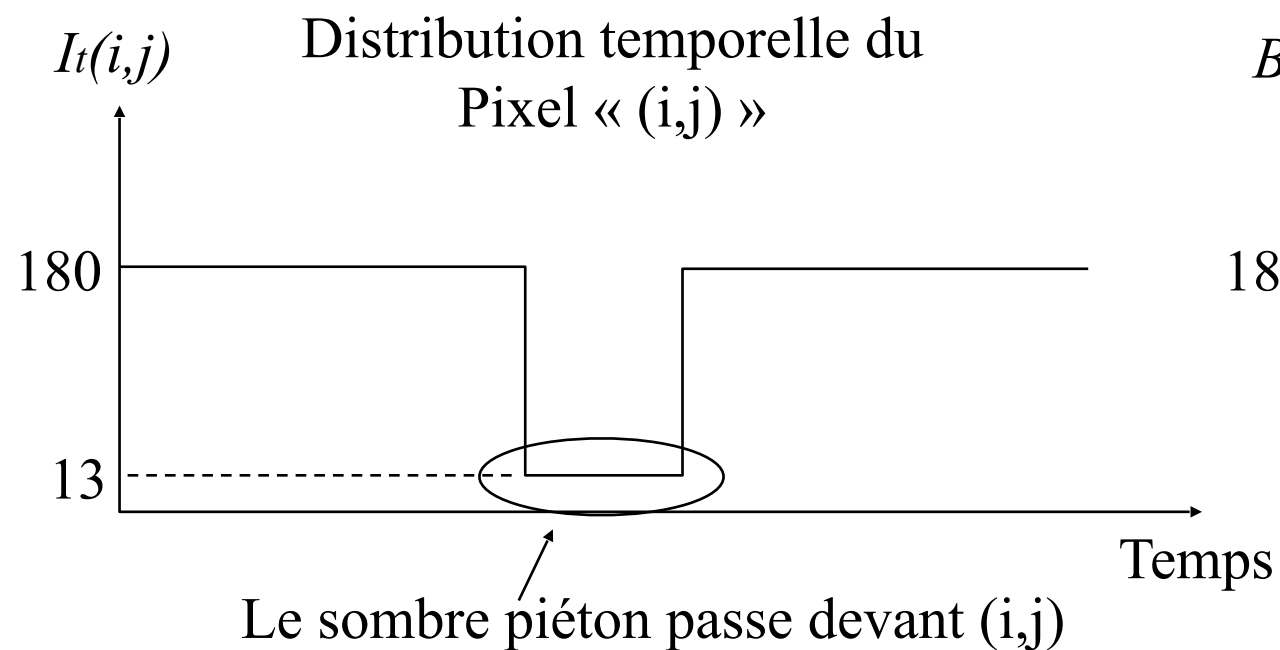


Opérations inter-images.

Détection de mouvement



Pixel « (i,j) »

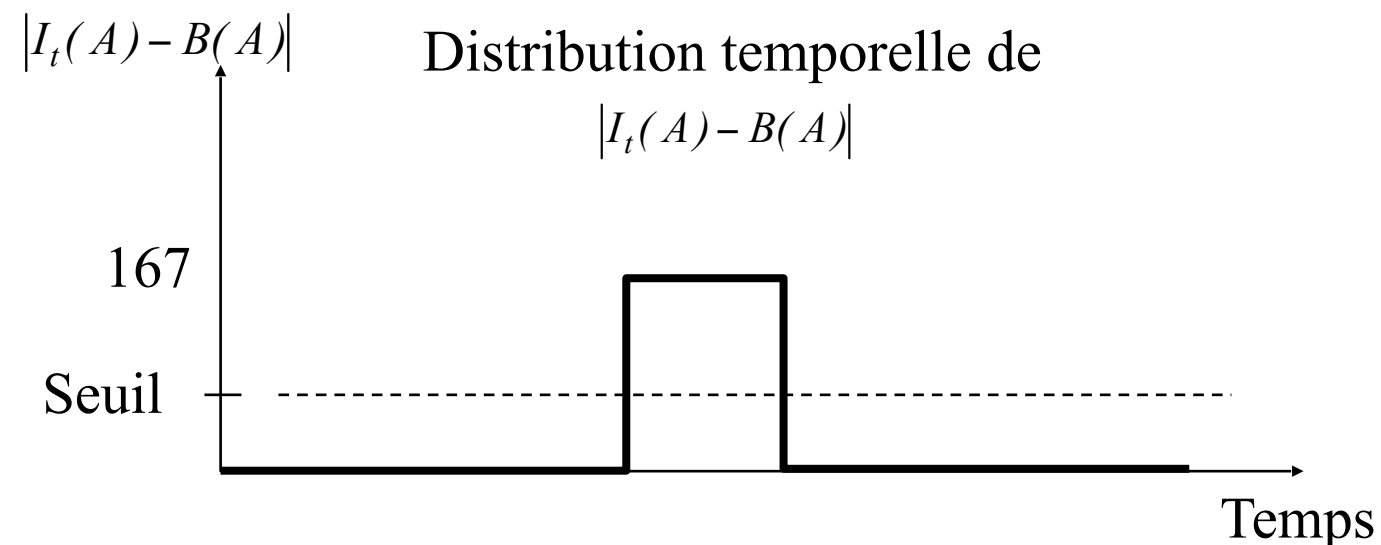
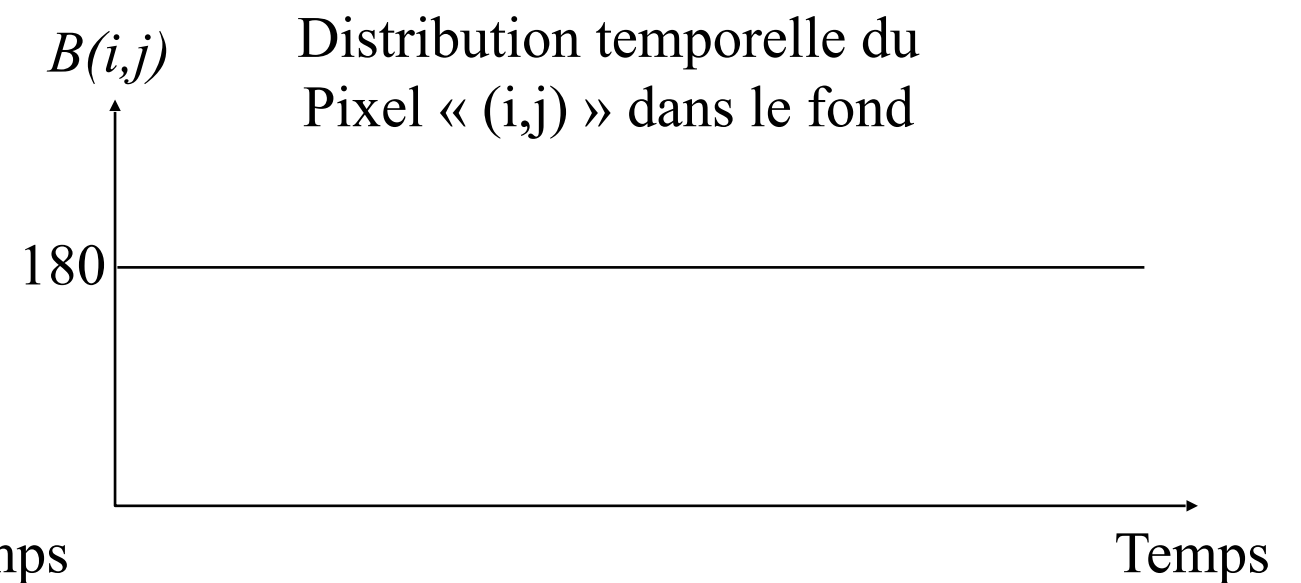
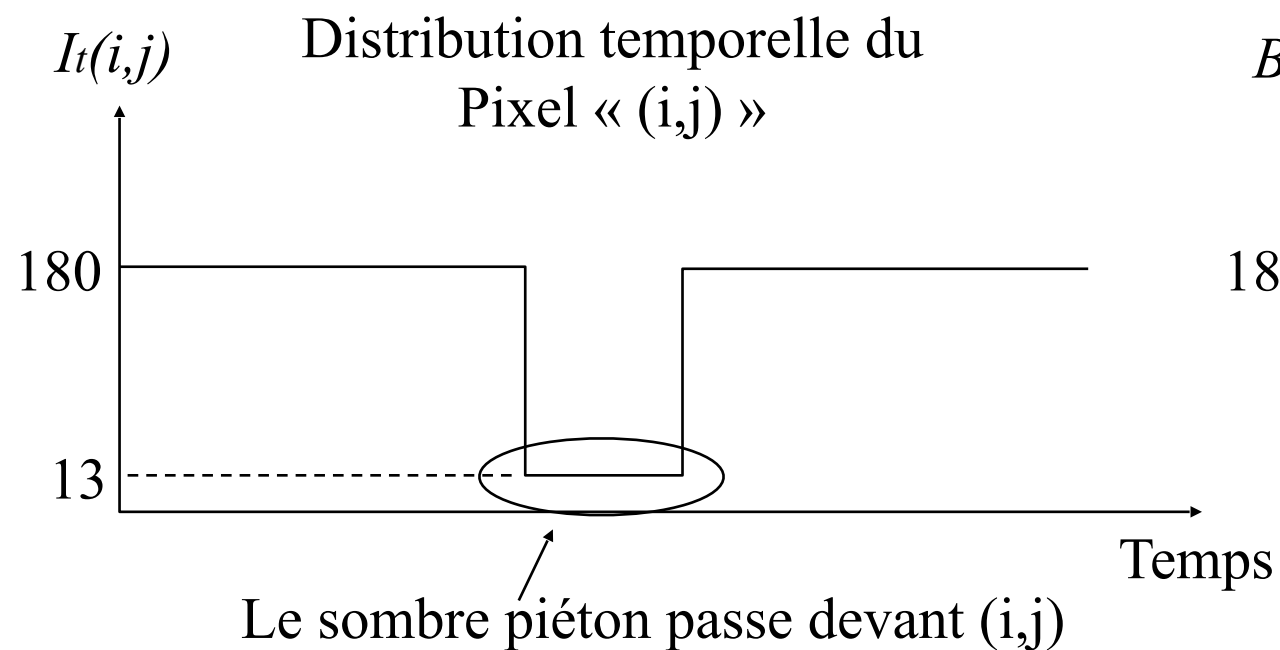


Opérations inter-images.

Détection de mouvement



Pixel « (i,j) »



Opérations inter-images.

Détection de mouvement



Pixel « (i,j) »

$$X_t(x, y) = \text{Seuil}(|I_t(x, y) - B(x, y)|)$$

Bien sûr, la méthode de soustraction de fond fonctionne bien *ssi* :

1. B est connu;
2. B est constant dans le temps;
3. I_t n'est pas une séquence « trop bruitée »;
4. les objets du fond sont immobiles;
5. la caméra est parfaitement fixe;
6. les objets en mouvement ont une couleur différente du fond.

Opérations inter-images.

Détection de mouvement

1. Que faire si B est inconnu? Il faut l'estimer!

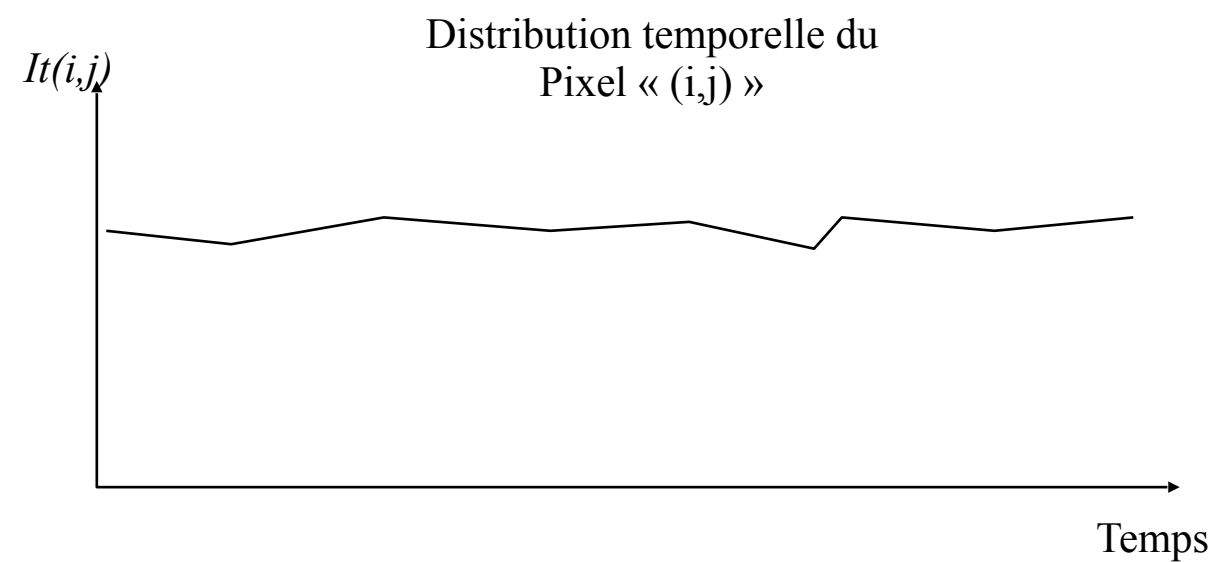


Pixel « (i,j) »

Opérations inter-images.

Détection de mouvement

1. Que faire si B est inconnu? Il faut l'estimer!

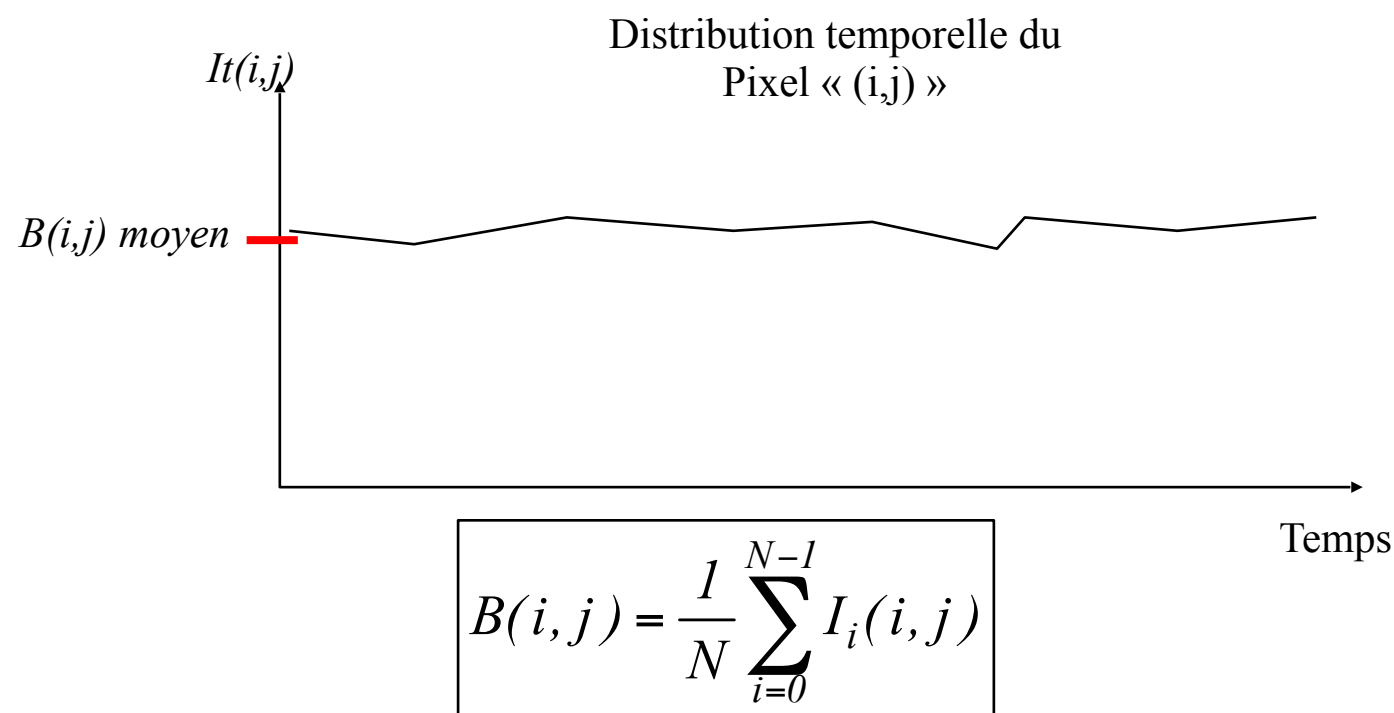


Pixel « (i,j) »

Opérations inter-images.

Détection de mouvement

1. Que faire si B est inconnu? Il faut l'estimer!



Pixel « (i,j) »

Pour estimer B à partir d'une séquence vidéo, il faut, pour chaque pixel de la séquence, distinguer la couleur/l'intensité du fond de la couleur des objets en mouvement et des erreurs de transmission. Pour ce faire, on peut

1. Prendre la moyenne
2. Prendre la médiane.

Opérations inter-images.

Détection de mouvement

1. Que faire si B est inconnu? Il faut l'estimer!



Pixel « (i,j) »

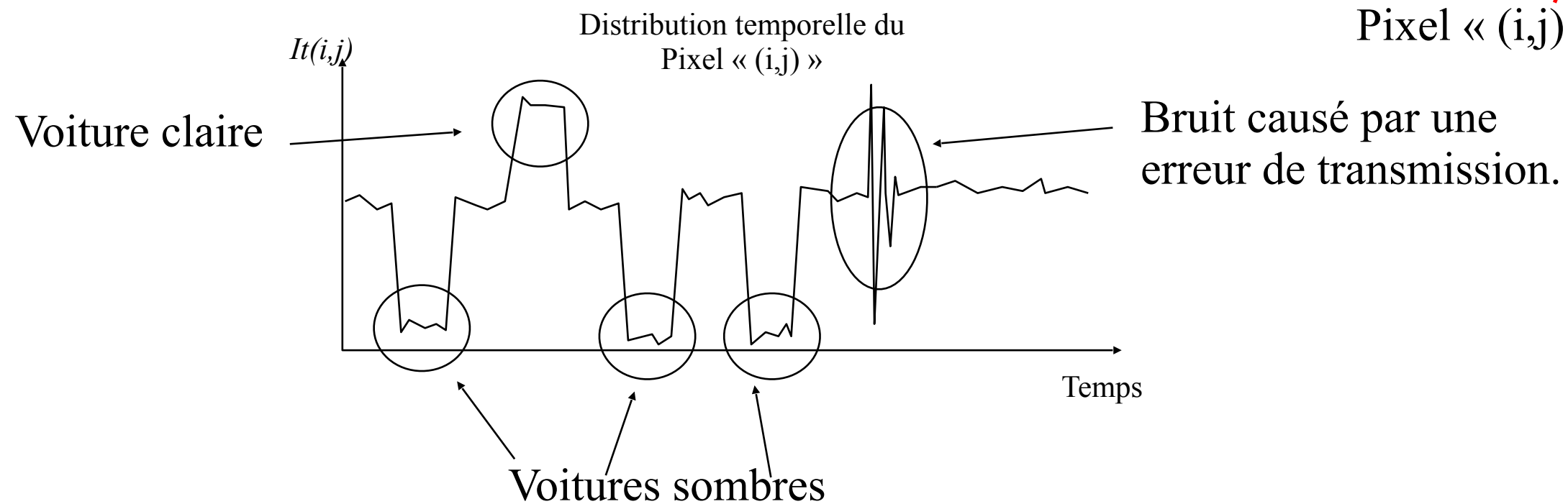
Opérations inter-images.

Détection de mouvement

1. Que faire si B est inconnu? Il faut l'estimer!



Pixel « (i,j) »



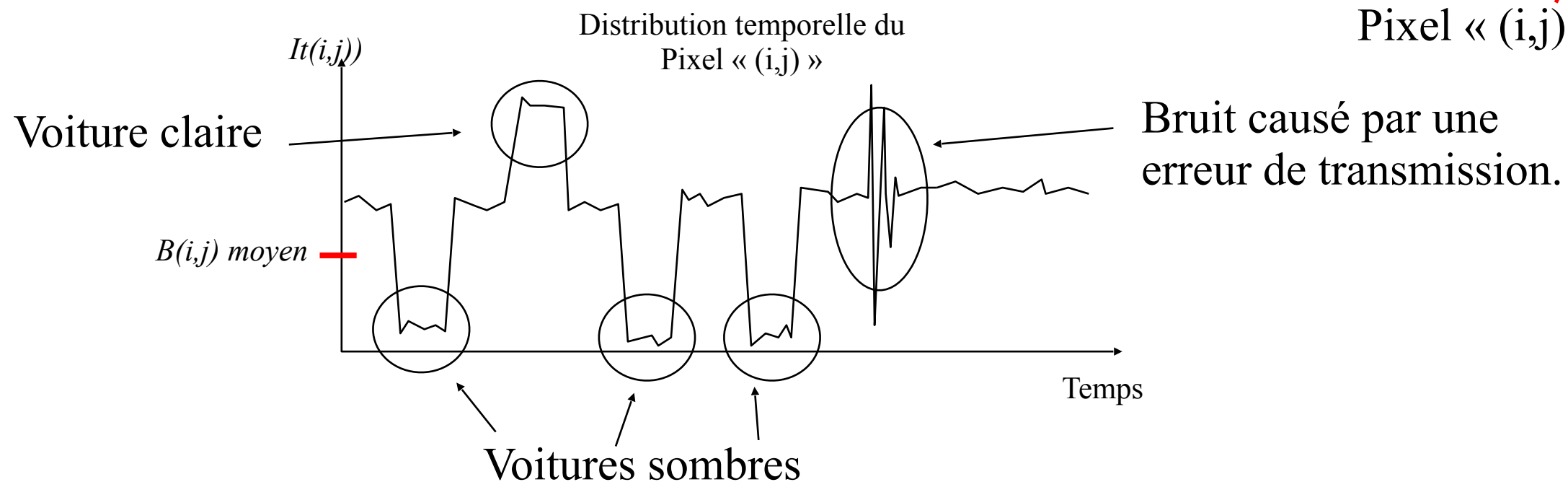
Opérations inter-images.

Détection de mouvement

1. Que faire si B est inconnu? Il faut l'estimer!



Pixel « (i,j) »



Moyenne:

$$B(i, j) = \frac{1}{N} \sum_{i=0}^{N-1} I_i(i, j)$$

ATTENTION : la moyenne ne fonctionne que si les observations $I_i(i, j)$ contiennent des valeurs se distribuant de façon **gaussienne** autour d'une valeur moyenne. Cette méthode est donc très sensible aux valeurs extrêmes (les *outliers*).

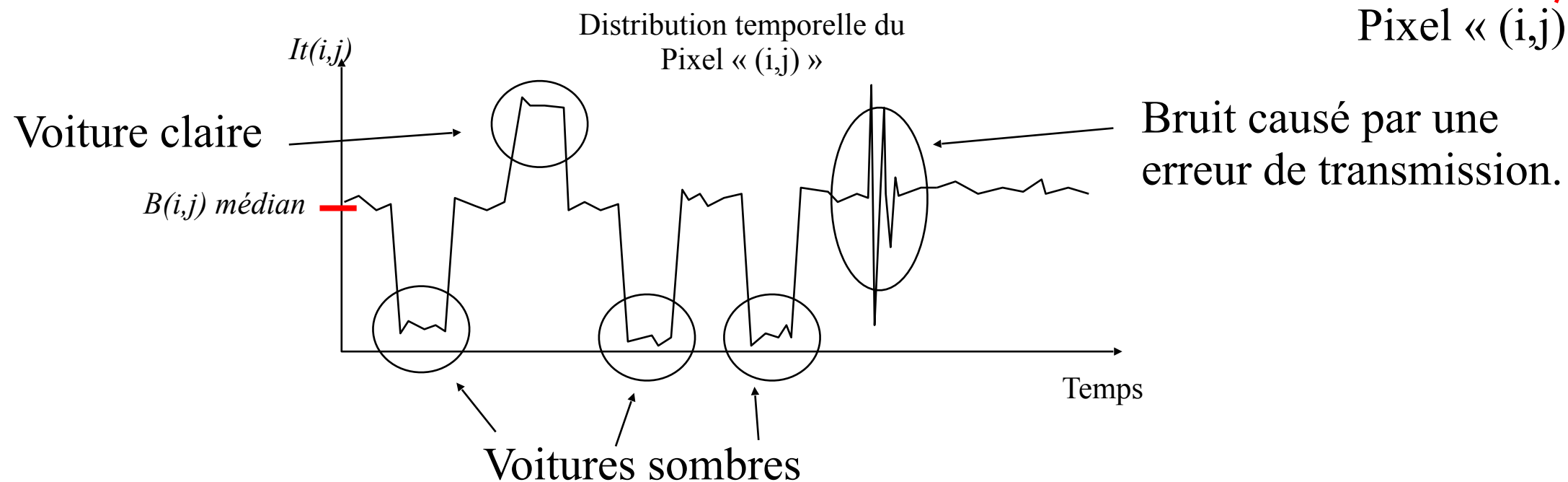
Opérations inter-images.

Détection de mouvement

1. Que faire si B est inconnu? Il faut l'estimer!



Pixel « (i,j) »



Médiane:

$$B(i, j) = \text{médiane}(I_0(i, j), I_1(i, j), I_2(i, j), \dots, I_{N-1}(i, j))$$

$$B(i, j) = \text{médiane}(I_0(i, j), I_{50}(i, j), I_{100}(i, j), \dots, I_{N-101}(i, j), I_{N-51}(i, j), I_{N-1}(i, j))$$

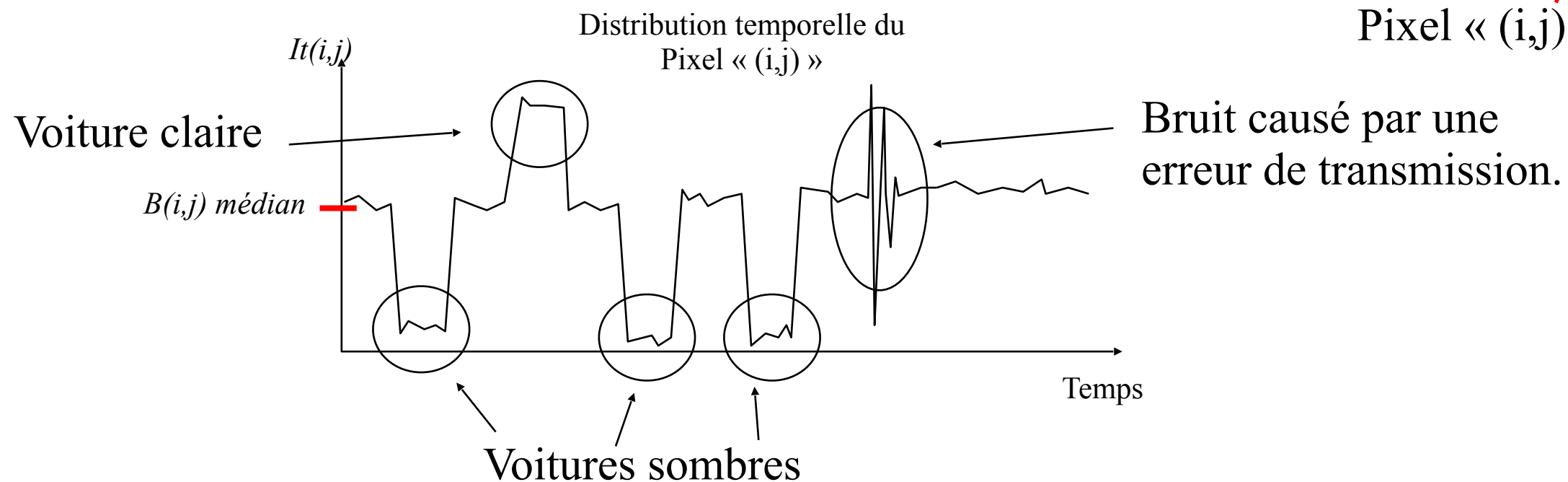
Opérations inter-images.

Détection de mouvement

1. Que faire si B est inconnu? Il faut l'estimer!



Pixel « (i,j) »



Médiane: $B(i, j) = \text{médiane}(I_0(i, j), I_1(i, j), I_2(i, j), \dots, I_{N-1}(i, j))$

$B(i, j) = \text{médiane}(I_0(i, j), I_{50}(i, j), I_{100}(i, j), \dots, I_{N-101}(i, j), I_{N-51}(i, j), I_{N-1}(i, j))$

ATTENTION : la médiane est robuste aux *outliers* à condition qu'ils ne correspondent pas à plus de 50% des observations.

Opérations inter-images.

Détection de mouvement



Médiane
temporelle



Médiane
temporelle



Opérations inter-images.

Détection de mouvement

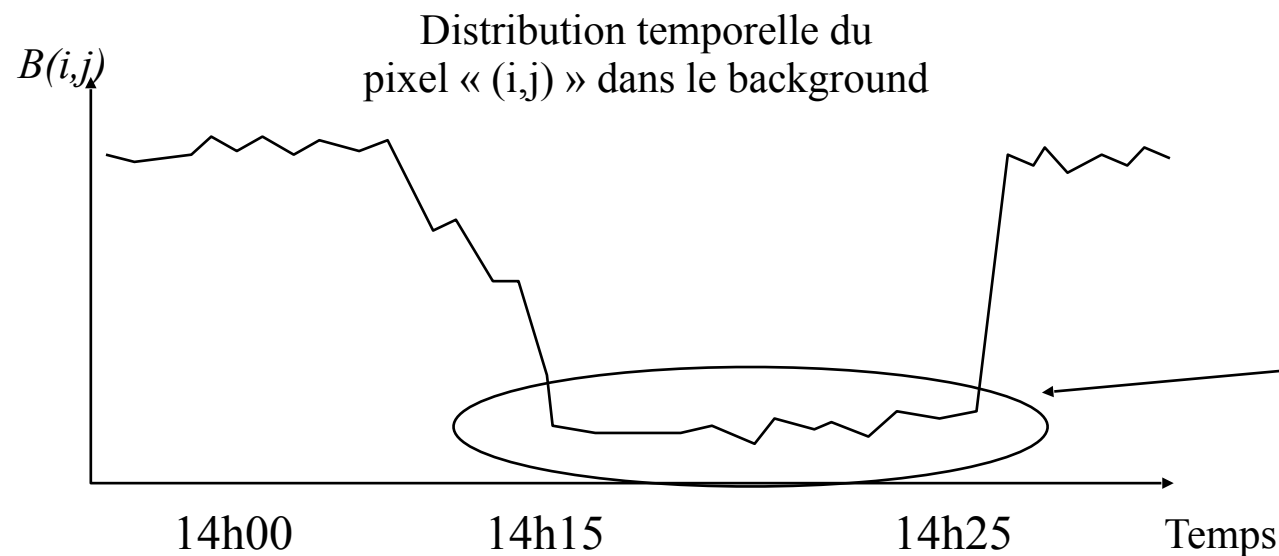
2. Que faire si B n'est pas constant dans le temps?



À 14h00 il faisait beau.



À 14h15 un gros nuage a obstrué le ciel.



Un nuage a momentanément obstrué le ciel.

Dans ce cas, il faut mettre à jour B à chaque temps t

$$B_t(i, j) = (1 - \alpha)B_{t-1}(i, j) + \alpha I_t(i, j)$$

où α est la « *forgetting constant* », généralement beaucoup plus petite que 1.

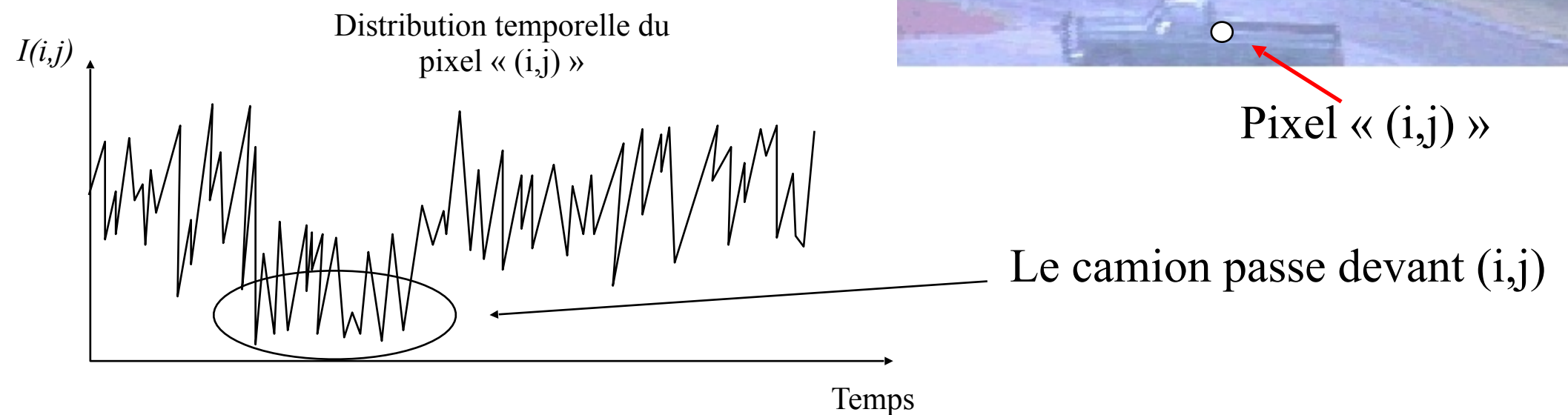
Afin d'éviter qu'un objet lent ne soit associé au fond, on peut faire ceci

$$B_t(i, j) = (1 - \alpha)B_{t-1}(i, j) + \alpha I_t(i, j) \quad \forall (i, j) \text{ t.q. } X_{t-1}(i, j) = 0$$

Opérations inter-images.

Détection de mouvement

3. Que faire si I est bruitée?



Le bruit peut causer de nombreux faux positifs et faux négatifs. Que pouvons-nous faire?

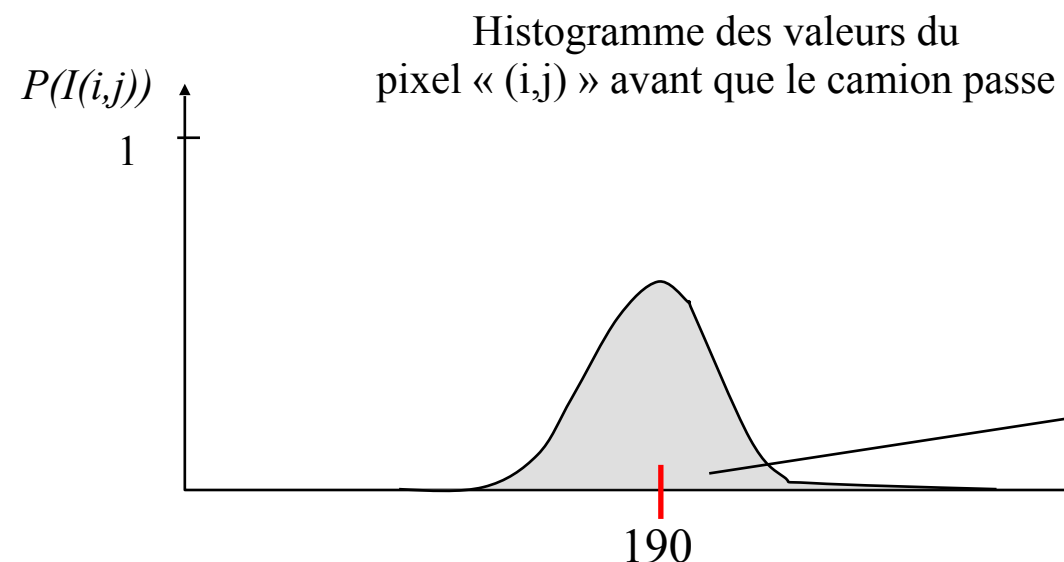
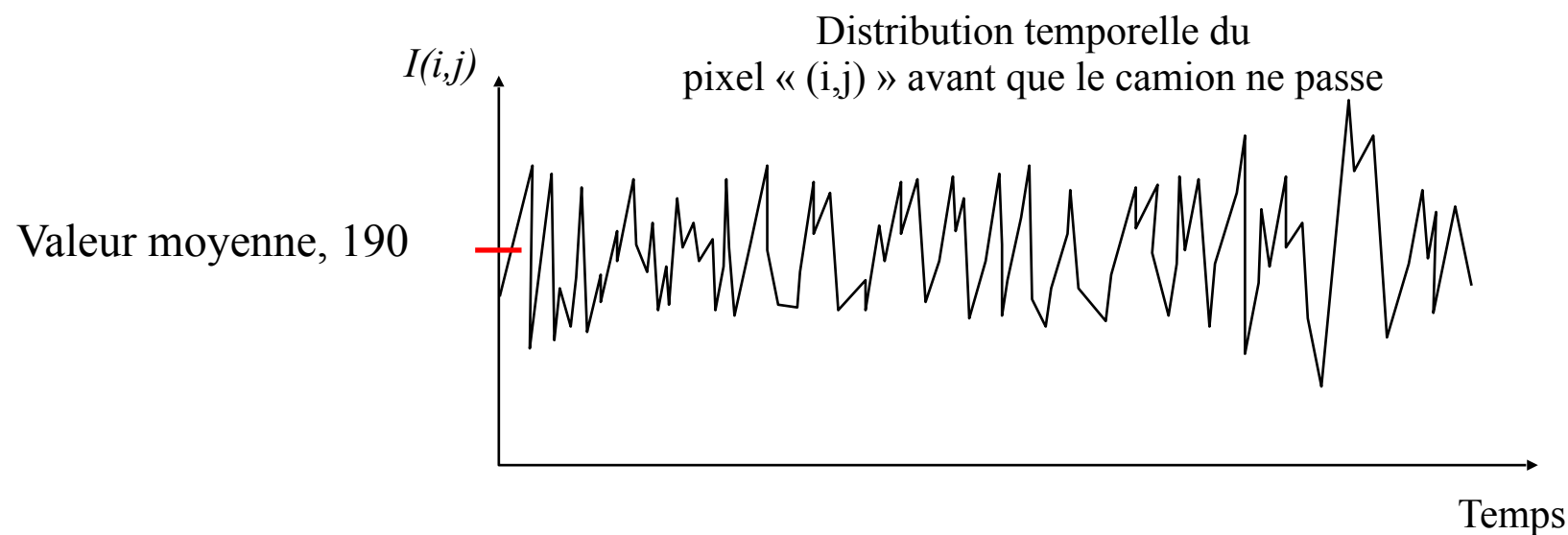
- 1- Puisque les faux positifs/négatifs forment un bruit impulsionnel, on peut utiliser un **filtre médian**.
- 2- On peut modéliser le fond de façon plus intelligente, c-à-d. de façon **probabiliste**.



Opérations inter-images.

Détection de mouvement

3. Que faire si I est bruitée?



Si le bruit est non corrélé dans le temps et de moyenne nulle, généralement l'histogramme des valeurs se distribuera suivant une « gaussienne ».

Gaussienne centrée sur 190 et d'écart type σ .

Opérations inter-images.

Détection de mouvement

3. Que faire si I est bruitée?

Au lieu de modéliser le pixel (i,j) par la valeur 190, on le modélisera par une gaussienne centrée sur 190 et d'écart type σ . Pour estimer la moyenne et l'écart-type de la gaussienne associée au pixel (i,j) , on a besoin d'une série de N frames sans mouvement, c'est-à-dire ne contenant que le fond. Dans ce cas,

$$\begin{aligned}\mu_{(i,j)} &= \frac{1}{N} \sum_{t=1}^{N-1} I_t(i,j) \\ \sigma_{(i,j)}^2 &= \frac{1}{N-1} \sum_{t=1}^{N-1} (I_t(i,j) - \mu_{(i,j)})^2\end{aligned}$$

Une fois $\mu_{(i,j)}$ et $\sigma_{(i,j)}$ estimés, la détection se fait à l'aide de l'opération suivante:

$$\begin{aligned}X_t(i,j) &= \text{Seuil}(P_{\mu,\sigma}(I_t(i,j))) \\ &= \text{Seuil}\left(\frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}\right)\end{aligned}$$

Opérations inter-images.

Détection de mouvement

les objets en mouvement ont la même couleur que le fond

C'est ce qu'on appelle un problème de « camouflage ». Malheureusement, il y a très peu de choses que l'on puisse faire avec une méthode par soustraction de fond. C'est une des limites inhérentes à cette approche. Toutefois, dans certains cas, on jouit d'une connaissance *a priori* du problème que l'on peut exploiter *a posteriori*.

Par exemple: Il me faut détecter des camions en mouvement. Puisque je sais qu'un camion est un objet plein, je peux remplir les trous (c-à-d éliminer les faux négatifs) à l'intérieur du véhicule une fois la détection faite.

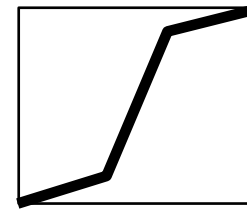


Les faits saillants

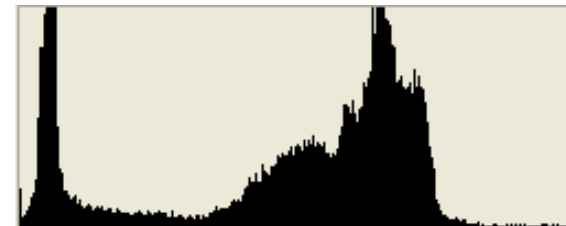
1. Correction gamma

$$g(x, y) = (f(x, y))^{\gamma}$$

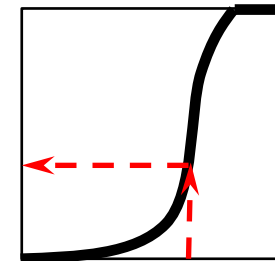
2. Transformations linéaires/non linéaires



3. Histogramme



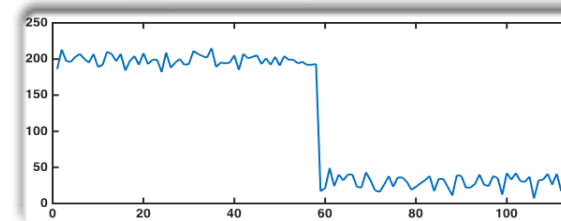
4. Égalisation d'histogramme



5. Warping & Interpolation



7. Bruit et rapport signal sur bruit



6. Détection de mouvement

