

Chapitre 2 : Domaine spatial

INF600F - Traitement d'images

Joël Lefebvre

UQÀM

INF600F - Automne 2024

Survol du cours

① Définitions et opérations de bases

② Opérations ponctuelles

③ Transformations géométriques

④ Fondements du filtrage spatial

⑤ Filtres non-linéaires

⑥ Démo Python

Announces

- À venir

Références

- (Chityala2020) Chapitre 4 : *Spatial Filters*
- (Burger2009, Vol1) Chapitre 4 : *Point Operations*
- (Burger2009, Vol1) Chapitre 5 : *Filters*
- (Gonzalez2018) Chapitre 3 : *Intensity Transformations and Spatial Filtering*

Rappel : Image numérique

<https://innovationsoftheworld.com/universite-du-quebec-a-montreal-uqam/>

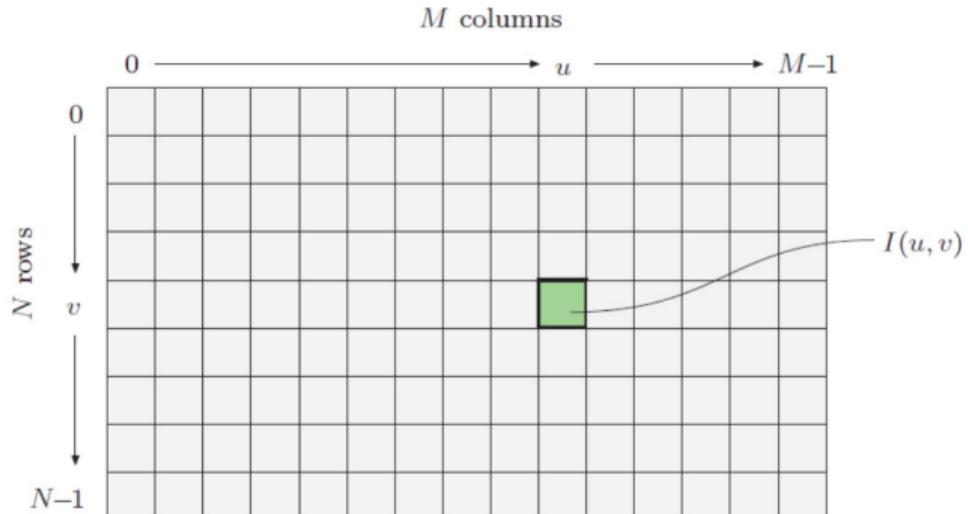
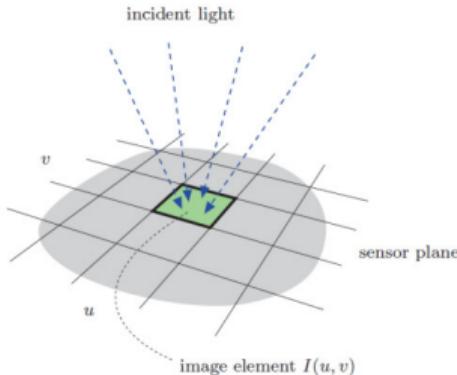


$F(x, y)$



148	123	52	107	123	162	172	123	64	89	...
147	130	92	95	98	130	171	155	169	163	...
141	118	121	148	117	107	144	137	136	134	...
82	106	93	172	149	131	138	114	113	129	...
57	101	72	54	109	111	104	135	106	125	...
138	135	114	82	121	110	34	76	101	111	...
138	102	128	159	168	147	116	129	124	117	...
113	89	89	109	106	126	114	150	164	145	...
120	121	123	87	85	70	119	64	79	127	...
145	141	143	134	111	124	117	113	64	112	...
:	:	:	:	:	:	:	:	:	:	:

$I(u, v)$



Source : (Burger, Vol1)

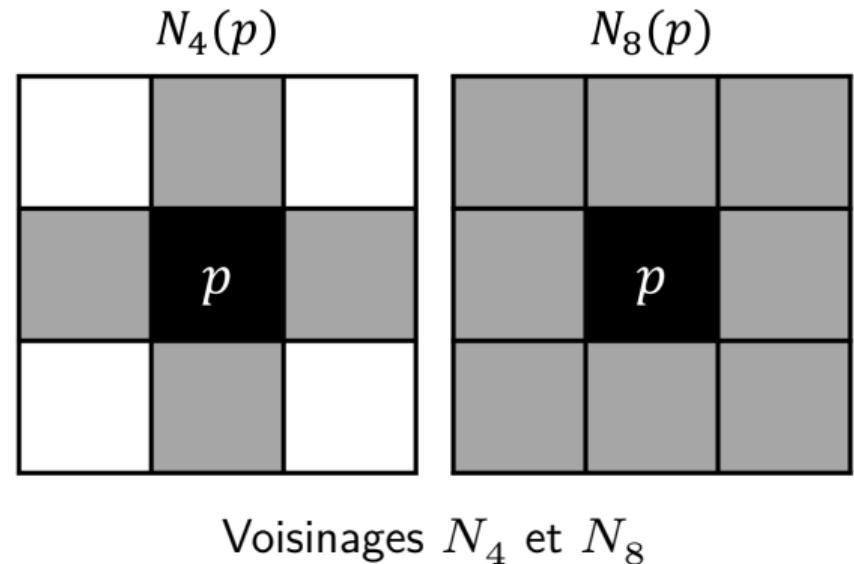
```
import imageio # Lecture / écriture des images
import numpy as np # Algèbre Linéaire
import matplotlib.pyplot as plt # Afficher images
```

Section 1

Définitions et opérations de bases

Opérations de base sur les pixels

- Voisinages
 - **Ordre 1** : 4 voisins $N_4(p)$
 - **Ordre 2** : 8 voisins $N_8(p)$
- Utilité de la notion de voisinage
 - Contours, Régions
 - Traitement spatial élémentaire (filtrage)
 - Modèles probabilistes d'images (Champs de Markov)



Connectivité du jeu démineur ?



Wikimedia

Adjacence, régions et frontières

- **Importance de ces notions** : définition rigoureuse de régions et contours pour les algorithmes de détection / traitement.
- **Adjacence** : deux pixels p et q sont adjacents si $q \in$ voisinage de p
 - **4-adjacence** : $q \in N_4(p)$
 - **8-adjacence** : $q \in N_8(p)$ (*plus faible*)
- **Chemin ou courbe** : Ensemble de pixels $\{p_i; 1 \leq i \leq n\}$ tels que
 $\forall i$: p_i et p_{i+1} adjacents

Définitions mathématiques

- $x \in A$: l'élément x appartient à l'ensemble A
- \forall : pour tout

Adjacence, régions et frontières (suite)

Connexité

- **Région** R : ensemble de pixels appartenant à une image
- R **connexe** : signifie que $\forall(p, q) \in \mathbb{R}^2$, il existe un chemin connexe de pixels de R permettant de joindre p à q
- Régions 4-connexes, ou 8-connexes

Frontière

- Soit R une région d'une image
- **Frontière** : pixels p de R adjacents à au moins un pixel de \bar{R}
- \bar{R} est l'ensemble des pixels n'étant pas dans la région R
- Le type d'adjacence (4 ou 8) doit être précisé.

Distance entre pixels (1)

- Distance entre **pixels** \neq distance entre **images**

Distances classiques

- Soit 2 pixels $p_1 : (x_1, y_1)$ et $p_2 : (x_2, y_2)$
- **Distance de Manhattan** (L_1)

$$D_1(p_1, p_2) = |x_2 - x_1| + |y_2 - y_1| \quad (1)$$

- **Distance euclidienne** (L_2)

$$D_2(p_1, p_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad (2)$$

- **Distance de Tchebychev** (*Chessboard*, L_∞)

$$D_\infty(p_1, p_2) = \max(|x_2 - x_1|, |y_2 - y_1|) \quad (3)$$

Distance entre pixels (2)

	3	2	3	
3	2	1	2	3
2	1	0	1	2
3	2	1	2	3
	3	2	3	

Distance de Manhattan
 $D_1 = 1$ est $N_4(p)$

2		2		2
	2	1	0	1
2		1		2
	2		2	
2		2		2

Distance Euclidienne

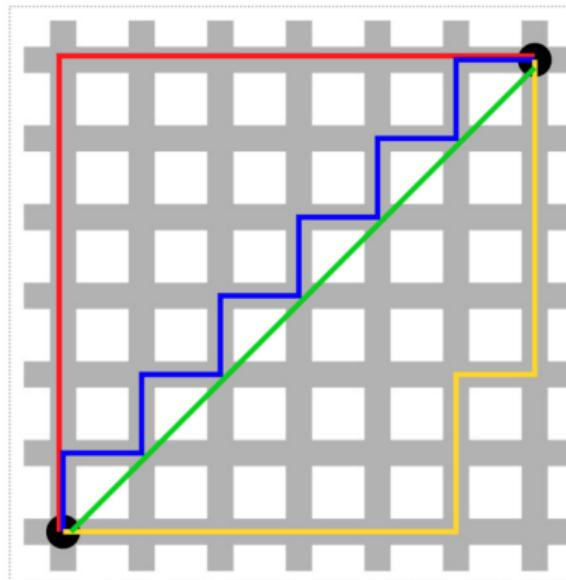
2	2	2	2	2
2	1	1	1	2
2	1	0	1	2
2	1	1	1	2
2	2	2	2	2

Distance Tchebychev
 $D_\infty = 1$ est $N_8(p)$

Quelques distances

Exemple : Distance entre pixels

- Quelle est la distance de chaque trajet ?



Sources : Quora et Wikimedia

Opérations sur les images

- **Précisions indispensables**

- Quantités mises en jeu
- Nature des opérations

- **Quantités en jeu**

- 2 images → 1 image (soustraction, multiplication ...)
- 1 image → 1 image (seuillage, filtrage, transformation ...)
- 1 image → 1 vecteur ou un scalaire (moyenne, variance, histogramme ...)

- **Nature des opérations**

- Principale distinction: linéaire vs non linéaire
- Linéarité : par rapport à quelle quantité ?

Opérations arithmétiques

- **2 images de même taille → 1 image**
- Appliquées **terme à terme** (pixel à pixel)
- Soient f et g deux images de **même taille**
 - **Addition** : $f(x, y) + g(x, y)$
 - **Soustraction** : $f(x, y) - g(x, y)$
 - **Multiplication** : $f(x, y) \times g(x, y)$
 - **Division** : $f(x, y)/g(x, y)$
- **Précautions indispensables**
 - Validité des opérations (ex. : une division par zéro)
 - Plage de variation du résultat

Exemple : Moyennage (addition)

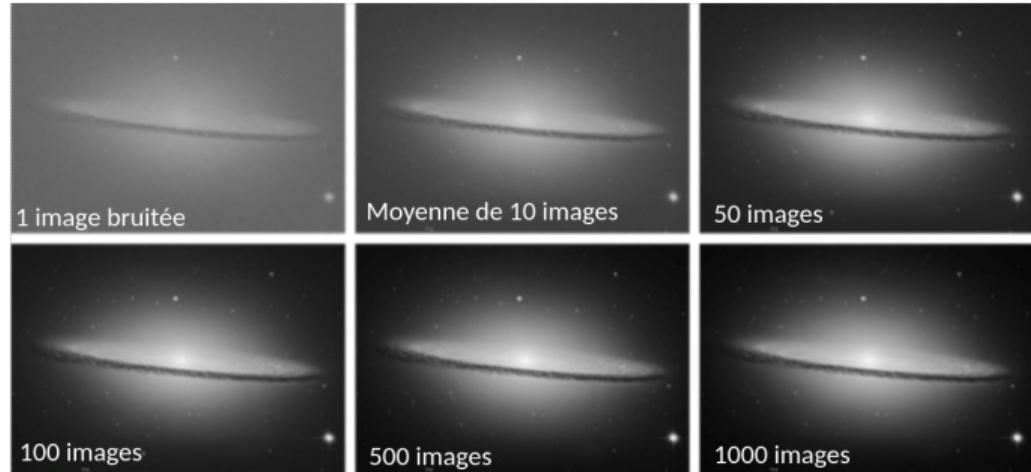
- **Application** : Débruitage
- **Image bruitée**

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

- **Restauration**

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

- où K est le nombre d'images



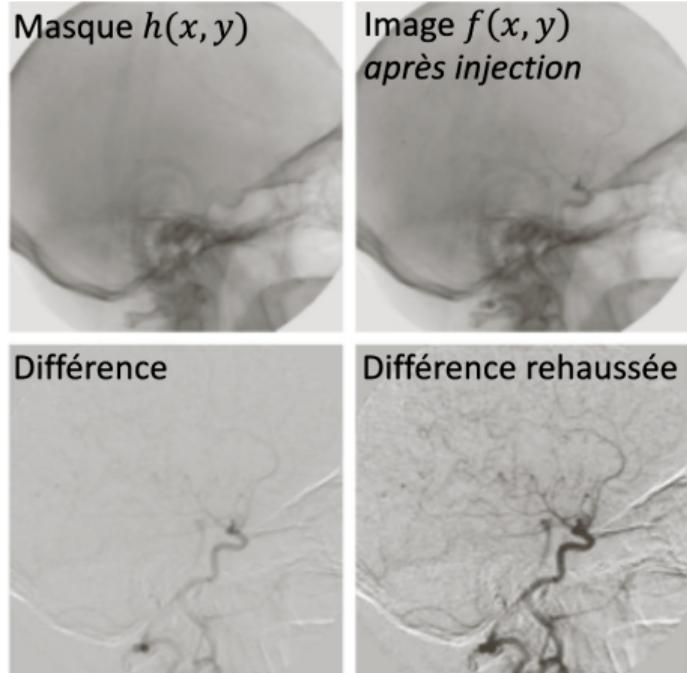
Galaxie Sombrero, à 31M années-lumière de la Terre
(Gonzalez, 2018)

Exemple : Imagerie différentielle

- **Application** : Angiographie par soustraction
- **Algorithm**e
 - Capture d'une radiographie $h(x, y)$
 - Injection intravasculaire d'un agent de contraste
 - Capture de radiographies avec une caméra rapide
 - Calcul des différences entre le masque $h(x, y)$ et chaque image acquise après injection $f(x, y)$

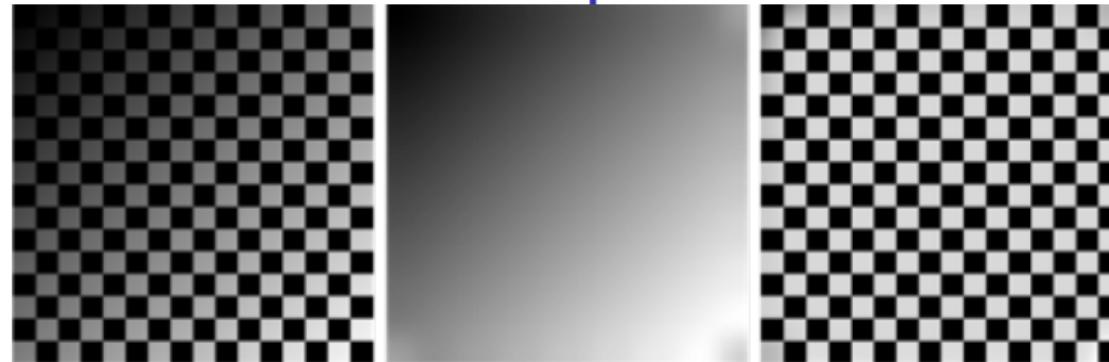
$$g(x, y) = f(x, y) - h(x, y)$$

- Rehaussement du contraste



(Gonzalez, 2018)

Exemple : Division et multiplication



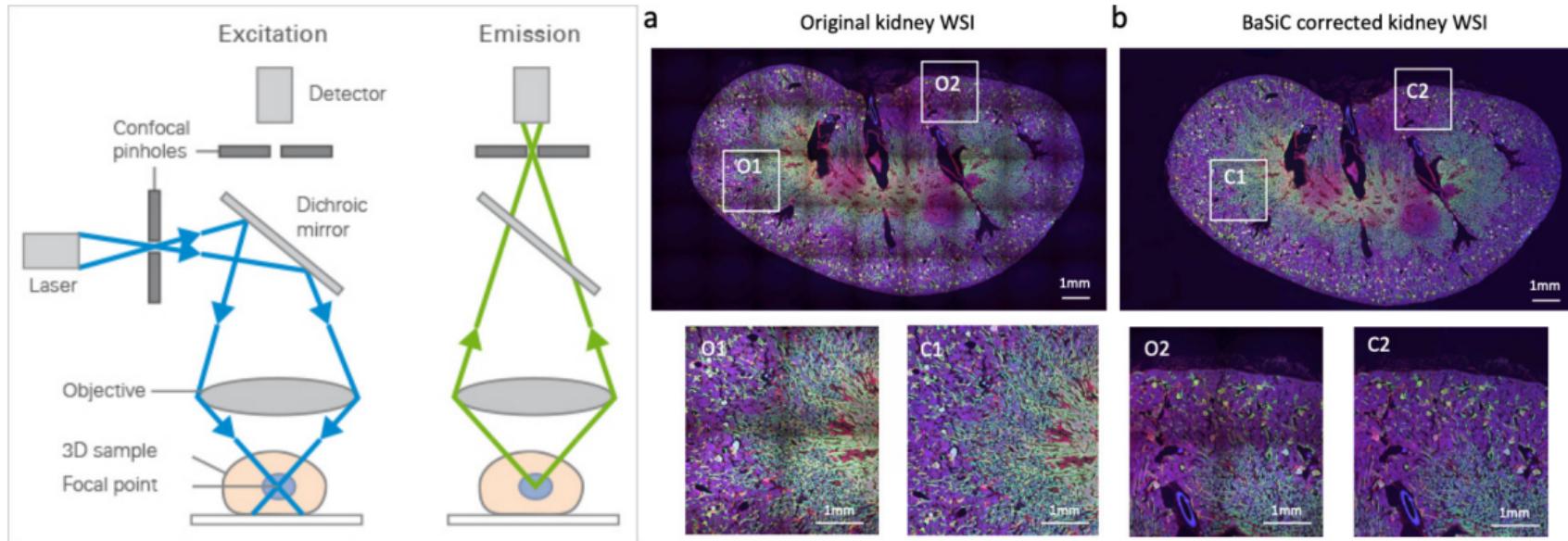
Correction d'une illumination non homogène par division (Gonzalez,2018)



Isolation des plombages par multiplication (Gonzalez,2018)

Exemple : Correction de l'illumination

- Correction de l'illumination pour la microscopie en mosaïque



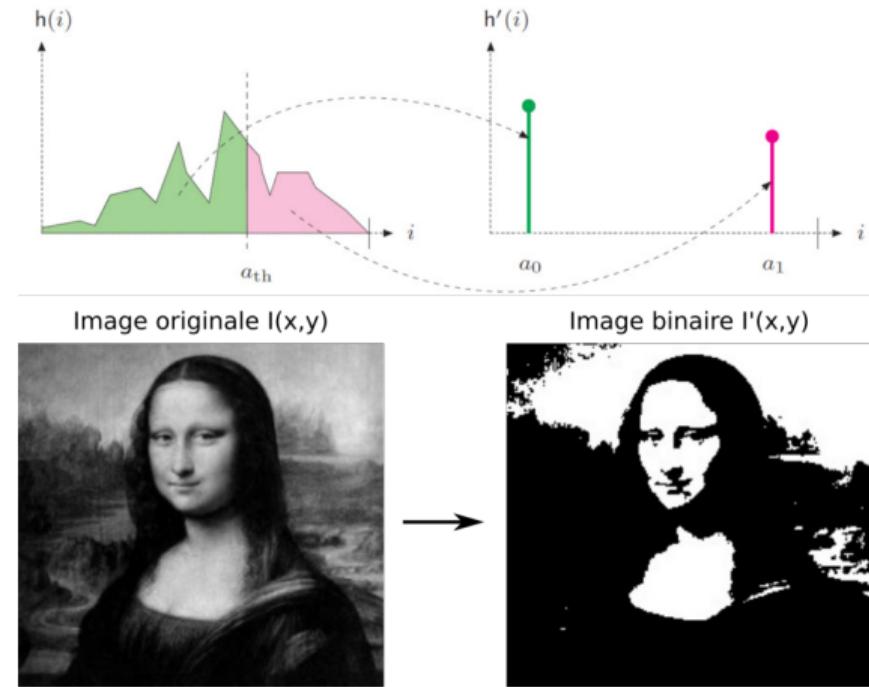
Microscope confocal ([microspedia](#)) et méthode BaSiC ([Peng et al \(2017\)](#))

Seuillage (*Thresholding*)

- Transforme une image grise en image binaire
- Le seuil d'intensité A est appliqué à tous les pixels

$$I'(x, y) = \begin{cases} 255 & I(x, y) > A \\ 0 & \text{sinon} \end{cases}$$

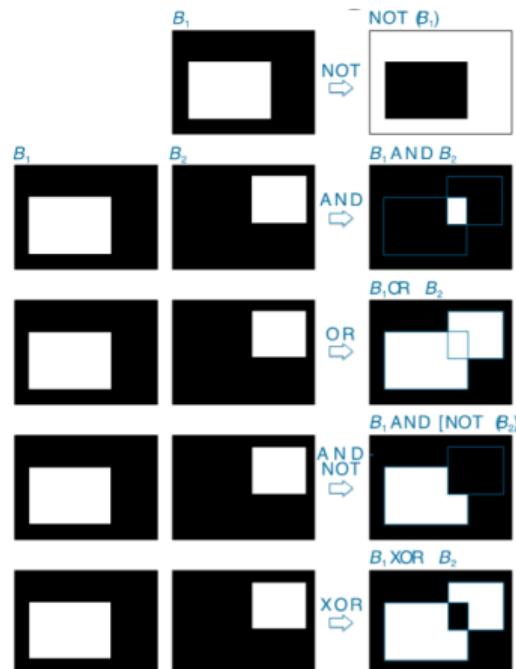
- Plusieurs autres méthodes de segmentation existent (*À venir*)



(Burger, 2009, Vol1)

Opérations logiques

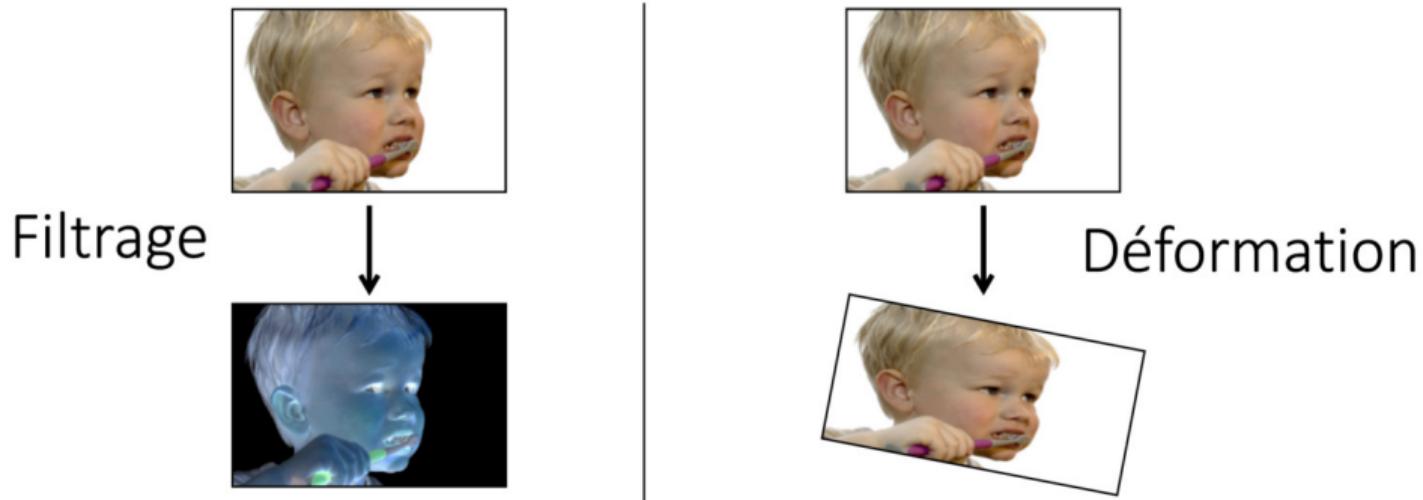
- **1 ou 2 images de même taille → 1 image**
- Image binaire (0/1)
- Principales utilisations
 - Utilisation de masques
 - Rehaussement
- Extension aux images non-binaires
- Soit $f(x, y)$ représentée sur k bits,
 $L = 2^k - 1$
 - Opérations “bit à bit”
 - $\bar{A} : L - f(x, y); (x, y) \in A$
 - Union (OR) → max
 - Intersection (AND) → min



(Gonzalez, 2018)

Transformations d'images dans le domaine spatial

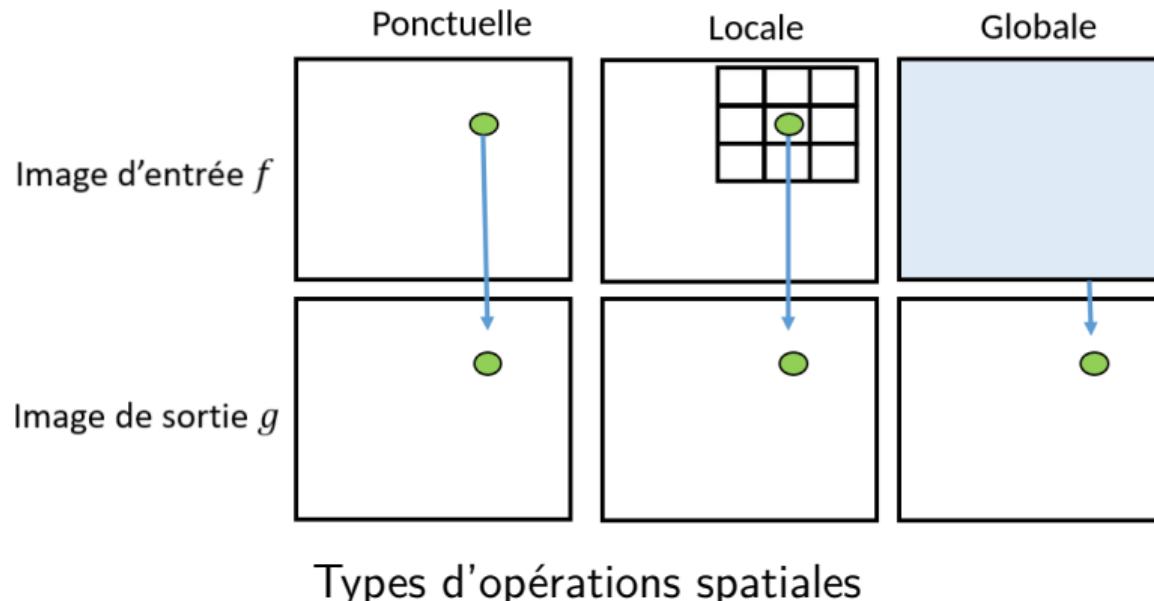
- **Quels types de transformations peut-on réaliser ?**
- **Filtrage** : Modification des valeurs des pixels $G(\vec{x}) = h(F(\vec{x}))$
- **Déformation** : Modification de la position des pixels $G(\vec{x}) = F(h(\vec{x}))$



Source : I. Gkoulekas

Types d'opérations spatiales

- **Ponctuelle** : portant sur les pixels “un à la fois”
- **Locale** : impliquant un voisinage du pixel
- **Globale** : impliquant l'ensemble de l'image

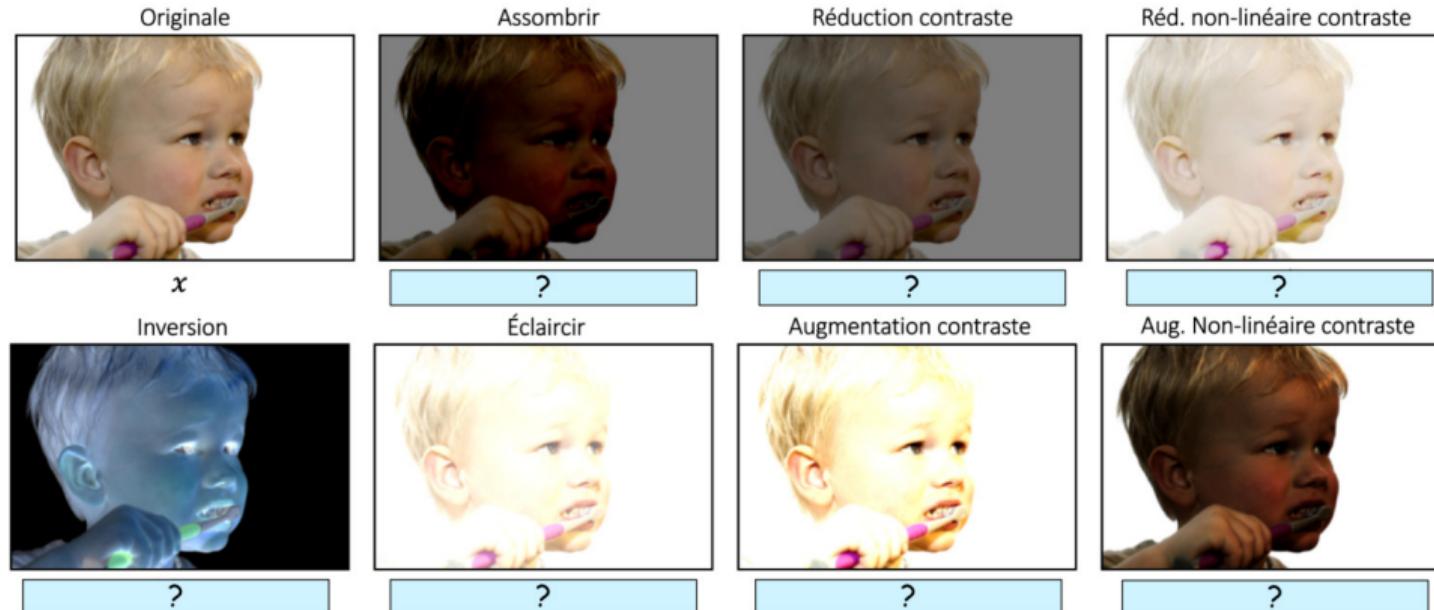


Section 2

Opérations ponctuelles

Exemples d'opérations ponctuelles (Questions)

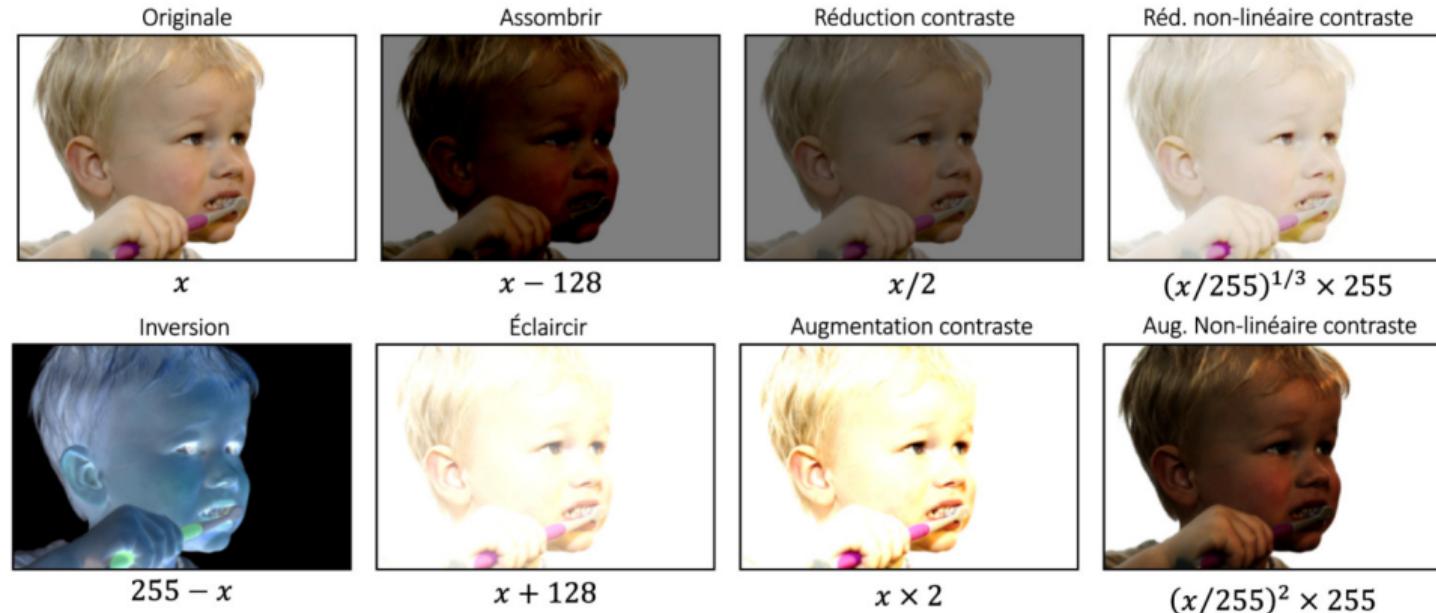
- **Comment implémenter ces opérations ?**



Source : I. Gkoulekas

Exemples d'opérations ponctuelles (Réponses)

- Comment implémenter ces opérations ?



Source : I. Gkoulekas

Plusieurs autres types d'opérations ponctuelles

[Bae et al., SIGGRAPH 2006]



Sortie de la caméra



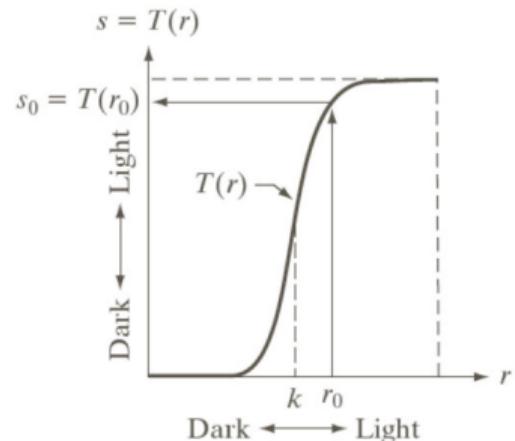
Image après une transformation stylistique (*tonemapping*)

Source : I. Gkoulekas

Transformations « pixel par pixel »

$$g(x, y) = T(f(x, y)) \quad (4)$$

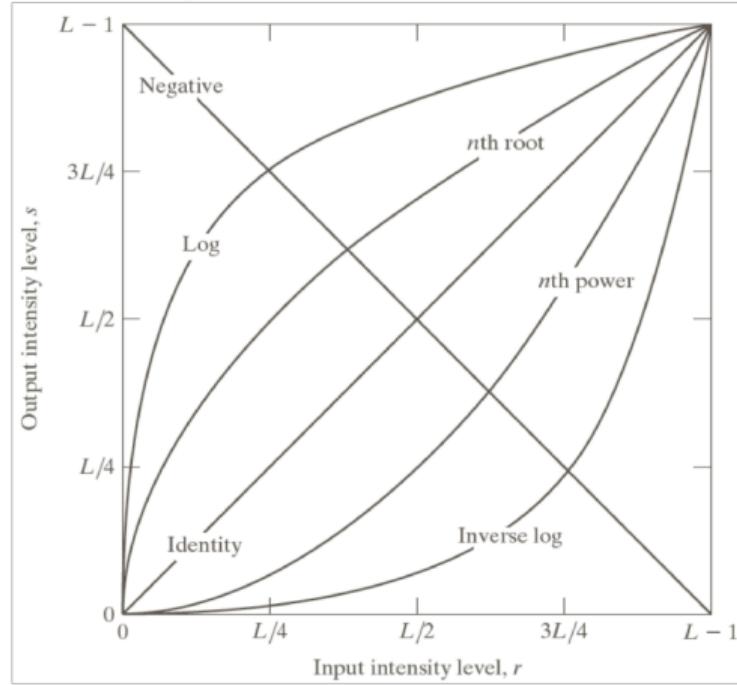
- (x, y) : Coordonnées d'un pixel
- $f(x, y)$: Intensité originale du pixel (x, y)
- $g(x, y)$: Intensité modifiée du pixel (x, y)
- $T(\cdot)$: Fonction de transformation des intensités
- **Choix de la fonction T ?**



(Gonzalez, 2018)

Transformations ponctuelles classiques

- Augmentation du contraste
- Inversion
- Transformation gamma
- Transformation logarithmique
- Transformation exponentielle
- Transformation *ad hoc*



(Gonzalez, 2018)

Transformations classiques

- **Objectifs**

- Ramener les intensités d'intérêt au centre de la plage de niveaux de gris
- “Étaler” la gamme des intensités d'intérêt

- **Précaution utile**

- Transformer le format de pixel en nombre réel (ex. : `numpy.float32`)
- Mettre à l'échelle l'image sur l'intervalle $[0, 1]$

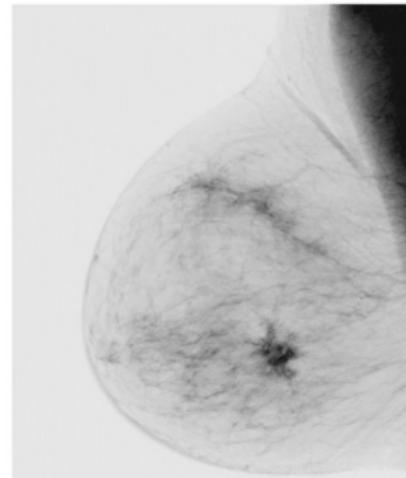
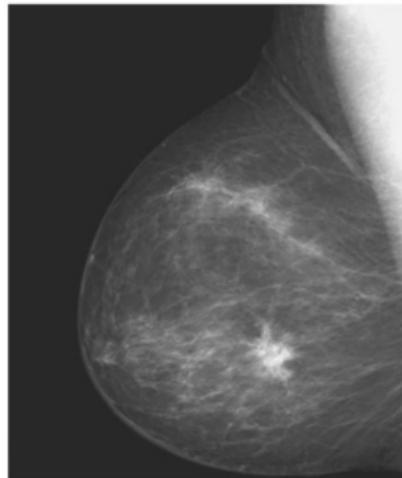
- **Principales transformations**

- **Inversion** : $T(r) = 1 - r$
- **Transformation gamma** : $T(r) = r^\gamma$
- **Transformation logarithmique** : $T(r) = \ln(1 + r) / \ln(2)$
- **Transformation exponentielle** : $T(r) = e^{r \ln(2) - 1}$

Inversion

$$T[f(x, y)] = 1 - f(x, y) \quad (5)$$

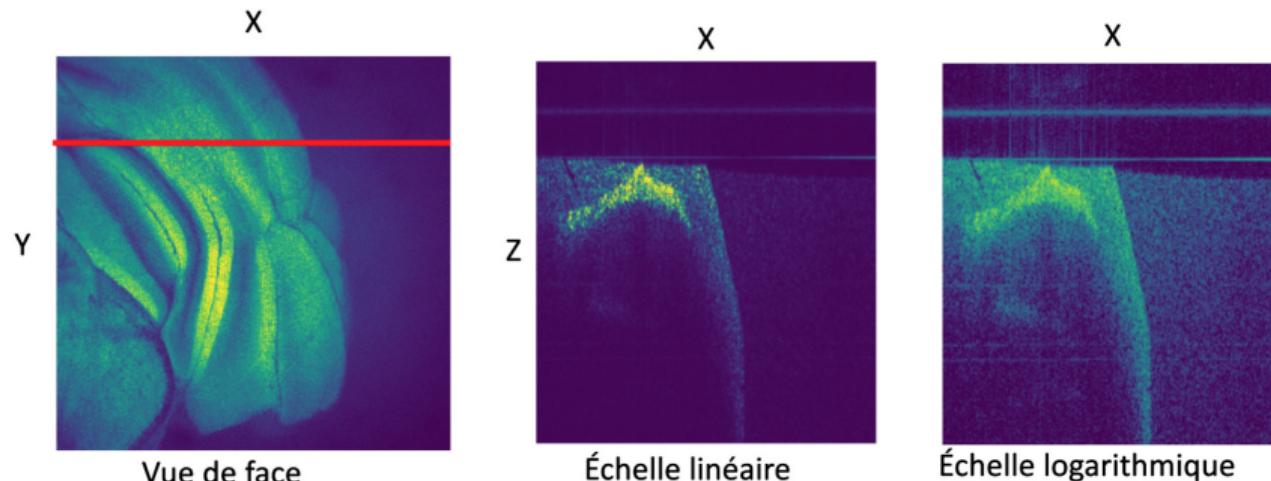
- pour $0 \leq f(x, y) \leq 1$
- **Exemple** : Négatif d'une mammographie



Transformation logarithmique

$$T[f(x, y)] = \log(f(x, y)) \quad (6)$$

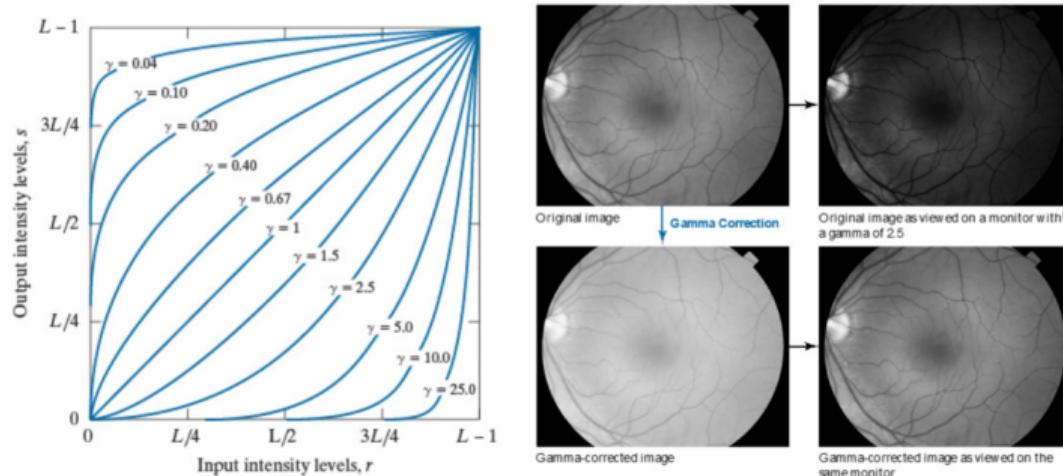
- Certaines modalités d'imagerie et de traitements d'image conduisent naturellement à des grandeurs exponentielles



Transformation gamma

$$T[f(x, y)] = cf^\gamma(x, y) \quad (7)$$

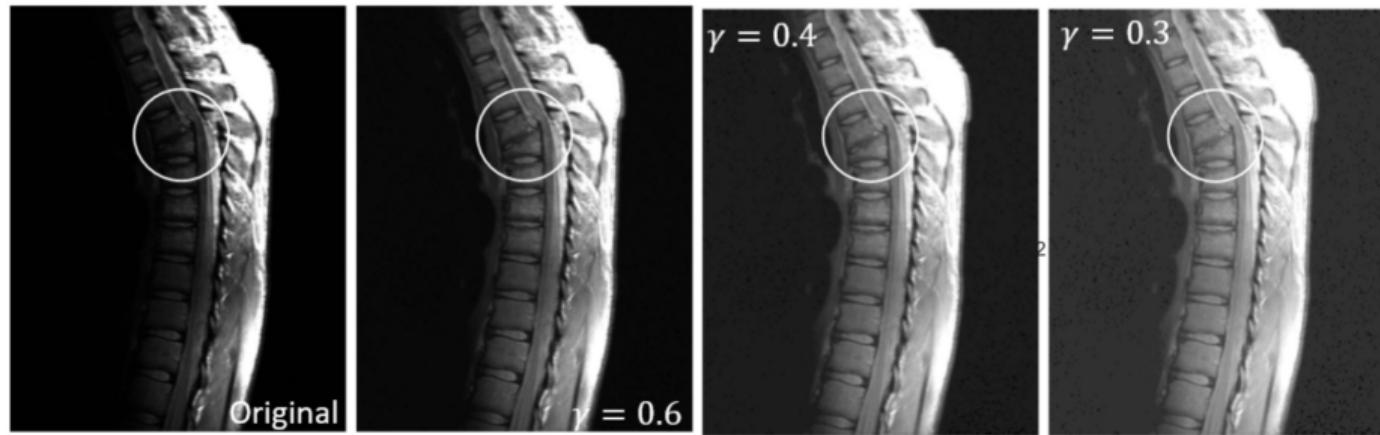
- **Origine** : Caractéristiques des moniteurs



(Gonzalez, 2018)

Exemple : IRM d'une colonne fracturée ($T(r) = cr^\gamma$)

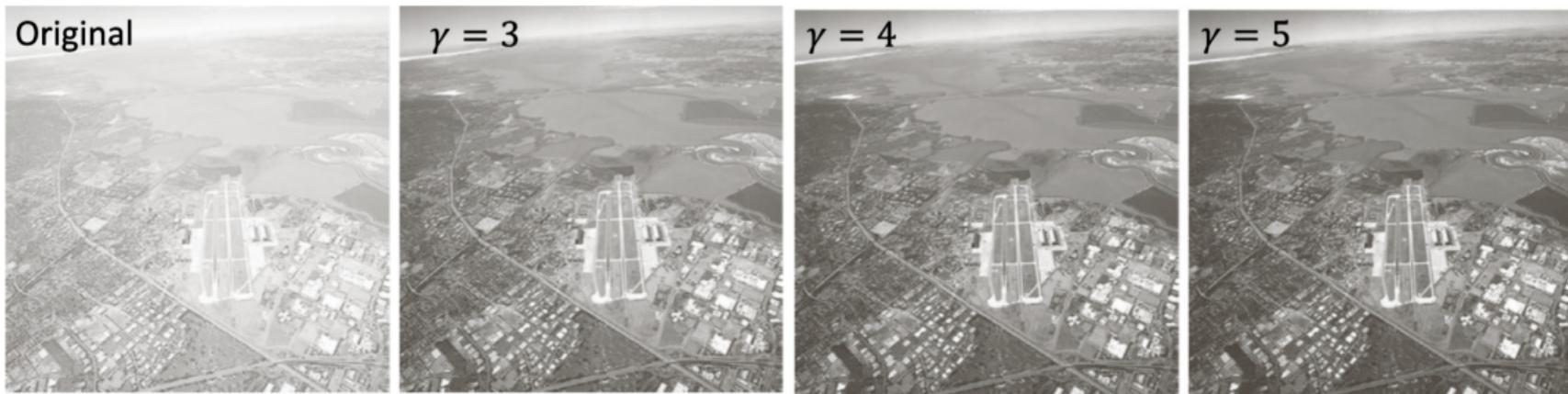
- **Image originale : trop sombre**
- Expansion des niveaux d'intensité désirable
- Accomplie avec un $\gamma < 1$
- Apparence améliorée due à un meilleur contraste



(Gonzalez, 2018)

Exemple : Imagerie aérienne ($T(r) = cr^\gamma$)

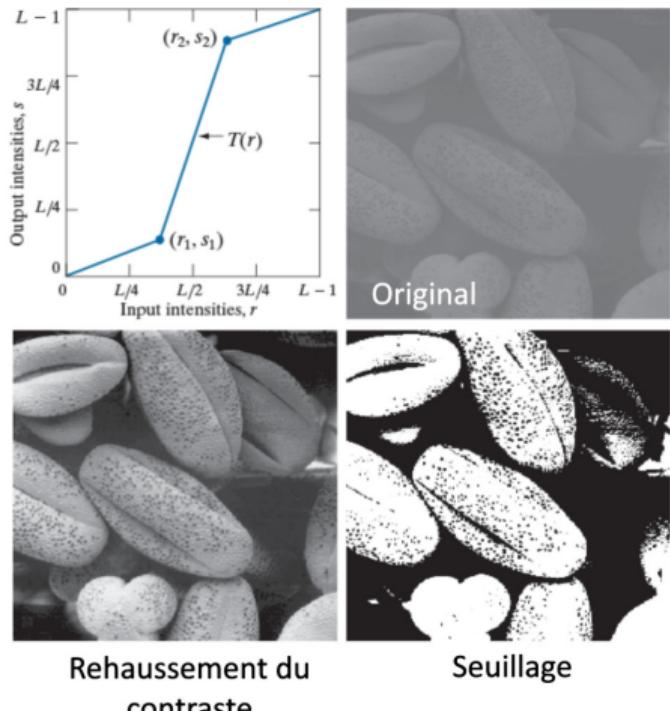
- **Image originale : Apparence délavée**
- Compression des niveaux d'intensité désirable
- Accomplie avec un $\gamma > 1$
- Apparence améliorée due à un meilleur contraste



(Gonzalez, 2018)

Transformation *ad hoc*

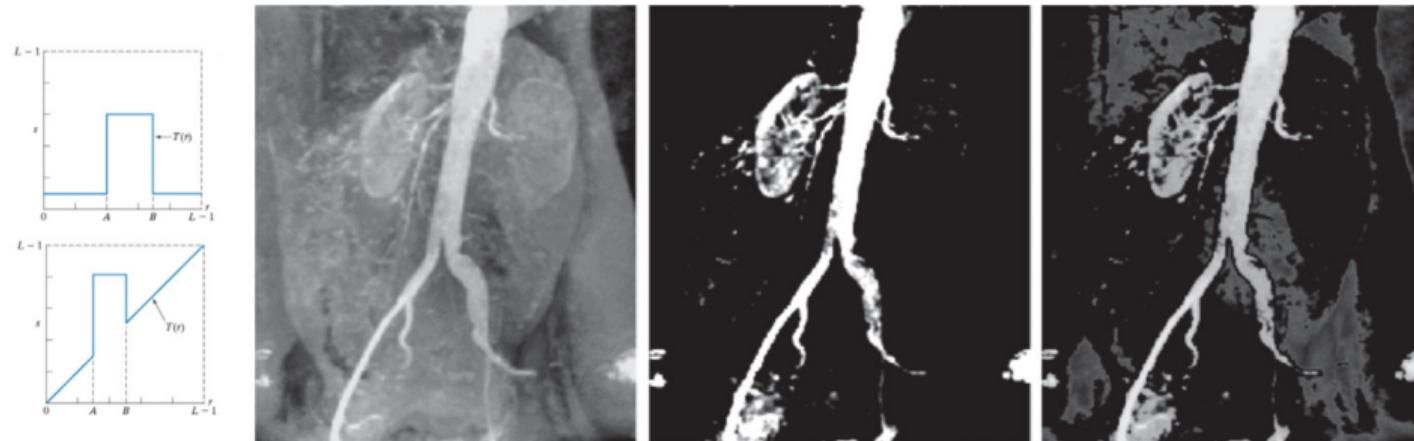
- Étirement du contraste
- Mise en évidence de plages particulières de niveaux de gris (*seuillage*)
- **Exemple** : Rehaussement d'une image de grains de pollen (700x) acquise par microscopie électronique.



(Gonzalez, 2018)

Exemple : Rehaussement d'une plage d'intensité

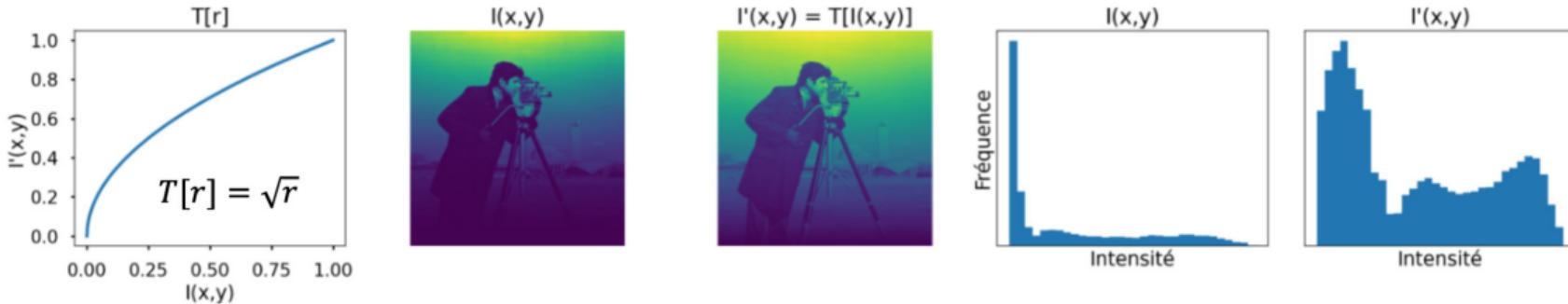
- **Technique** : Découpage des niveaux d'intensité (*slicing*)
- **Modalité** : Angiogramme de l'aorte près des reins



(Gonzalez, 2018)

Dénominateur commun

- Quelle est l'approche commune entre toutes les méthodes classiques de transformation de l'intensité ?
- Manipulation de l'histogramme
- Y a-t-il une approche générale et automatique ?



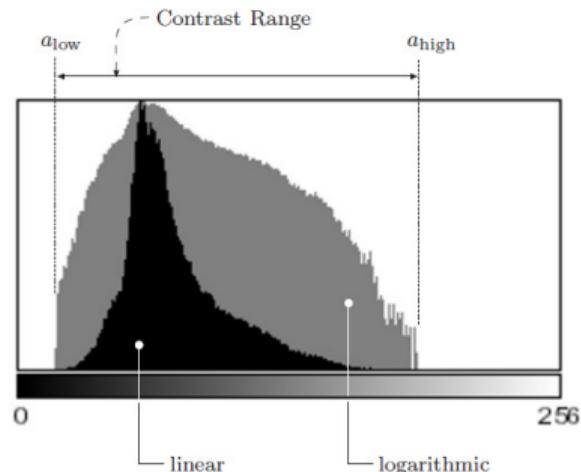
Effet d'une transformation sur l'histogramme

Histogramme d'une image

- Soit une image de taille (M, N) définie sur L niveaux r_k , $0 \leq k \leq L - 1$
- L'histogramme est calculé par

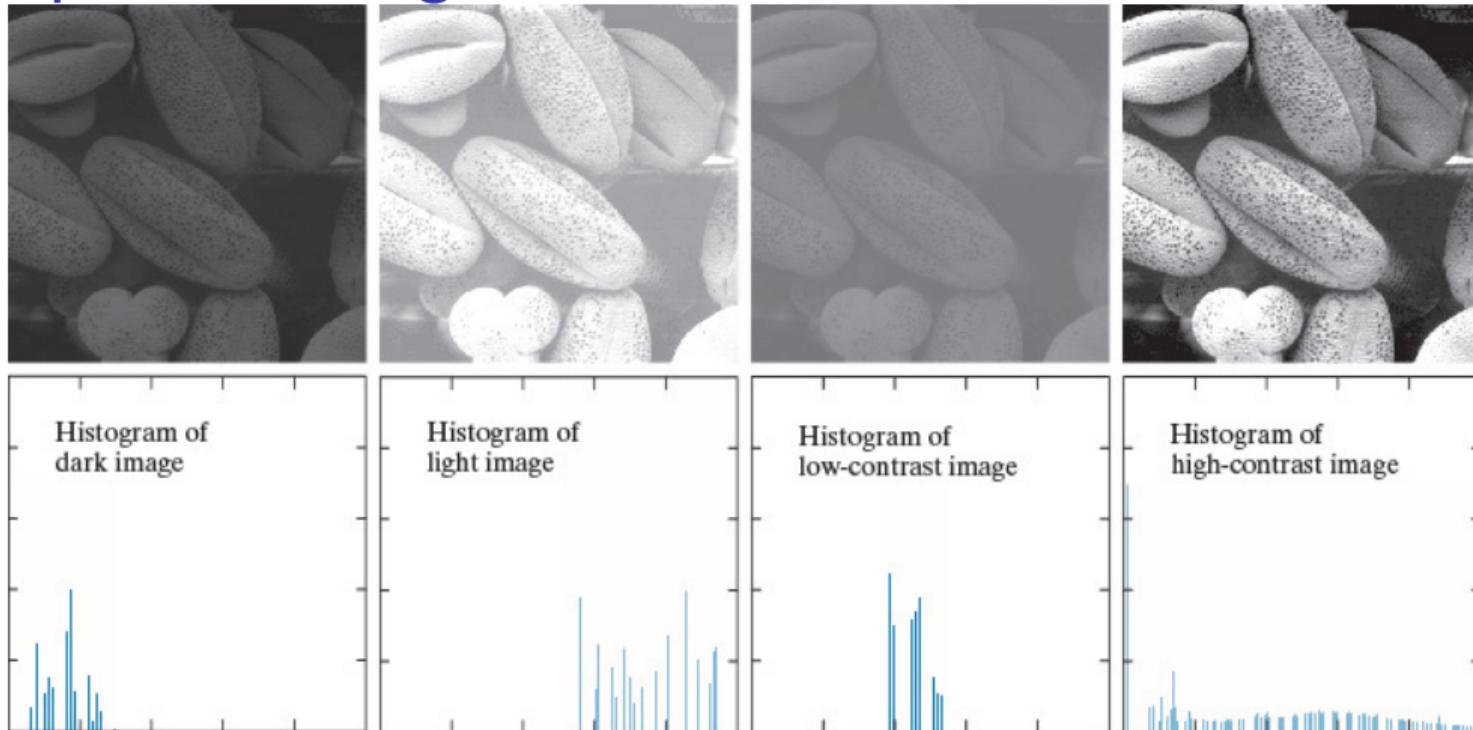
$$p(r_k) = \frac{n_k}{MN} \quad (8)$$

- où n_k est le nombre de pixels de valeur r_k
- **Caractéristiques désirables**
 - Couverture de tous les niveaux
 - Distribution uniforme \rightarrow Égalisation d'histogramme



(Burger, 2009, Vol1)

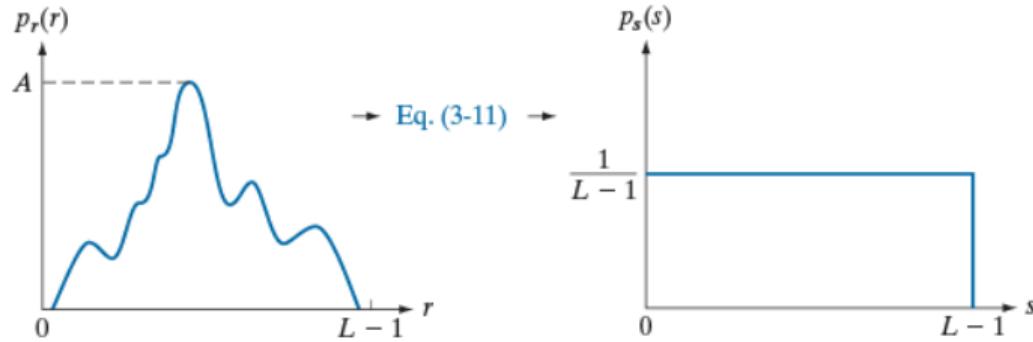
Exemples d'histogrammes



(Gonzalez, 2018)

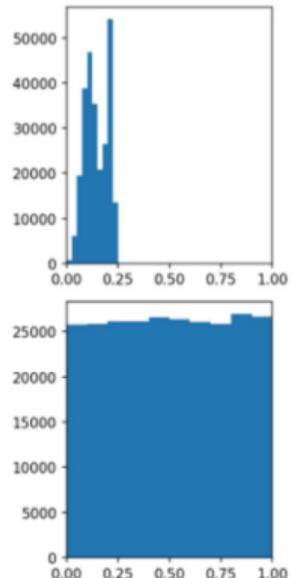
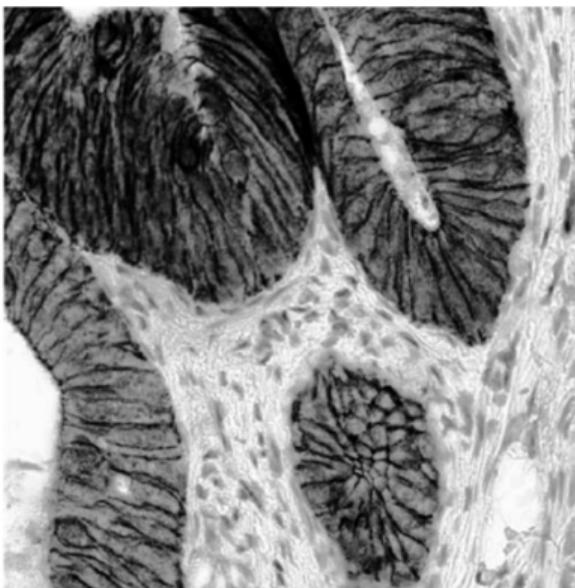
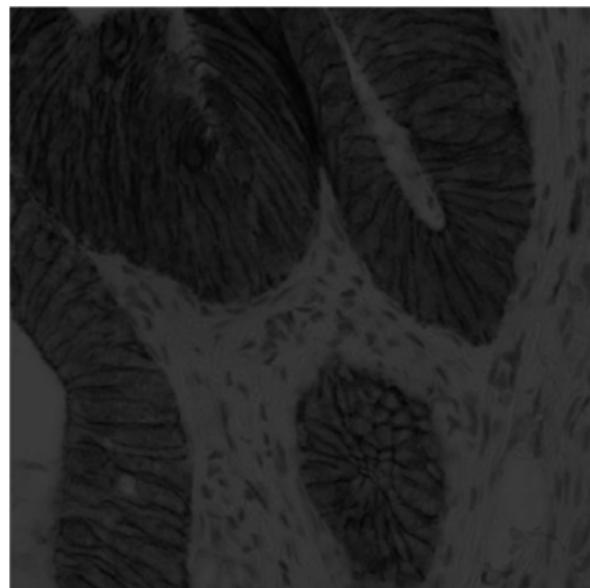
Égalisation d'histogramme

- Ajustement automatique de l'histogramme
- Transformation $s = T[r]$ telle que l'histogramme de l'image transformée soit plat
- **Exemple** : skimage.exposure.equalize_hist
- **En pratique**
 - Réduction du nombre de niveaux d'intensité
 - Efficacité variable



Exemple : Égalisation d'histogramme

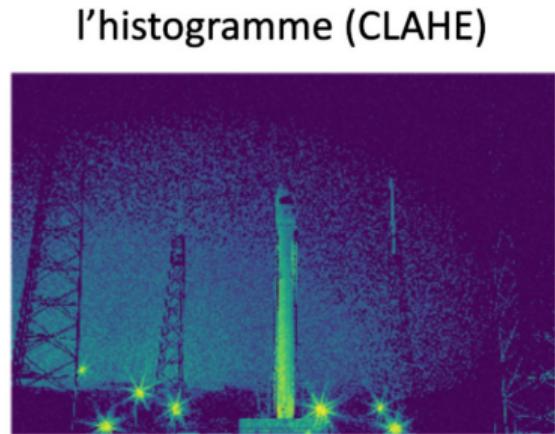
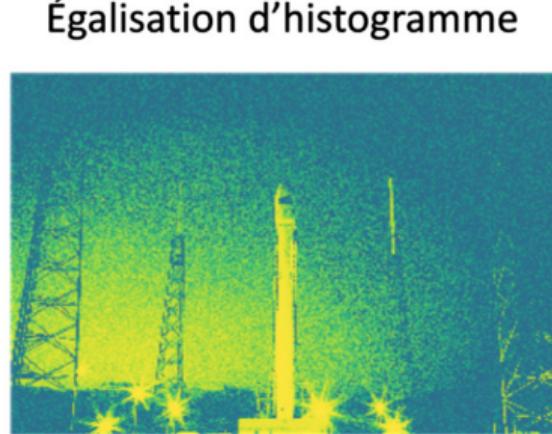
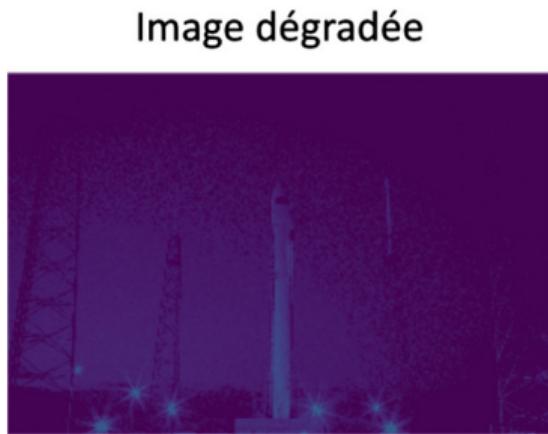
- Sous-exposition et ajustement automatique du contraste
- Utile si toute l'image est affectée de la même façon



Augmentation automatique du contraste d'une histologie

Adaptation locale de l'histogramme

- Utile si plusieurs régions d'intensités différentes
- Utilise l'histogramme local
- **CLAHE** : *Adaptation locale de l'histogramme limitée en contraste*
- Pour éviter de suramplifier le bruit dans les régions uniformes



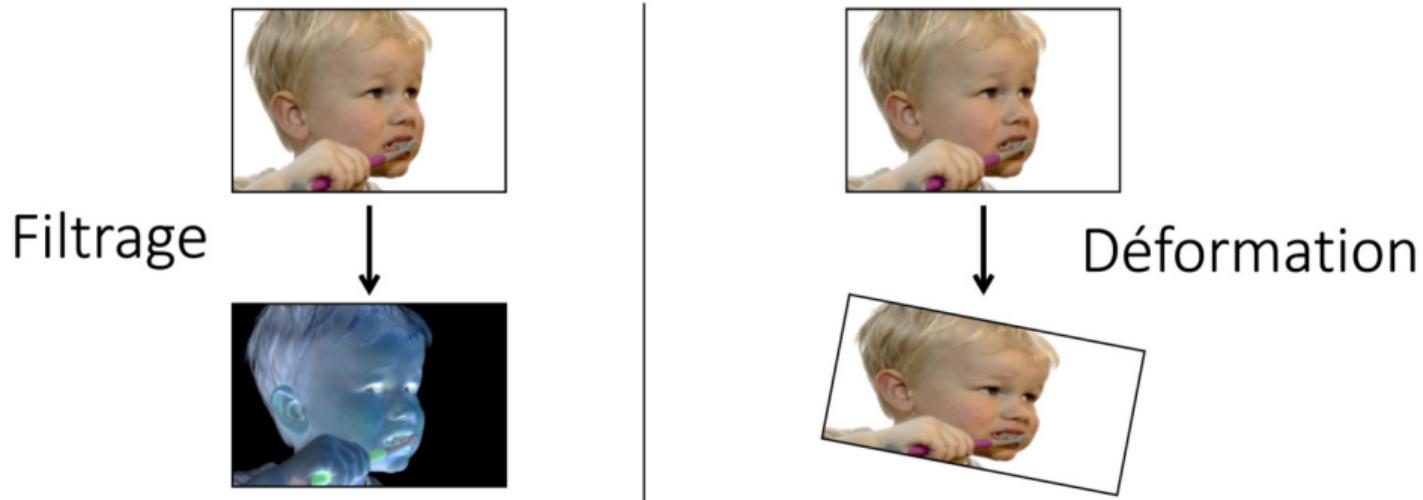
Exemple de correction locale de l'histogramme

Section 3

Transformations géométriques

Transformations d'images dans le domaine spatial

- **Quels types de transformations peut-on réaliser ?**
- **Filtrage** : Modification des valeurs des pixels $G(\vec{x}) = h(F(\vec{x}))$
- **Déformation** : Modification de la position des pixels $G(\vec{x}) = F(h(\vec{x}))$



Source : I. Gkoulekas

Transformations géométriques

- 1 image → 1 image de taille possiblement différente
- Transformation portant sur les coordonnées
- L'image transformée n'a pas nécessairement la même taille que l'image de départ
- Cas particulièrement important : *Transformations affines*

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \mathbf{A} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (9)$$

- où \mathbf{A} est la matrice affine (*transformation*). Elle peut représenter des rotations, translations, homothéties, cisaillement, etc.

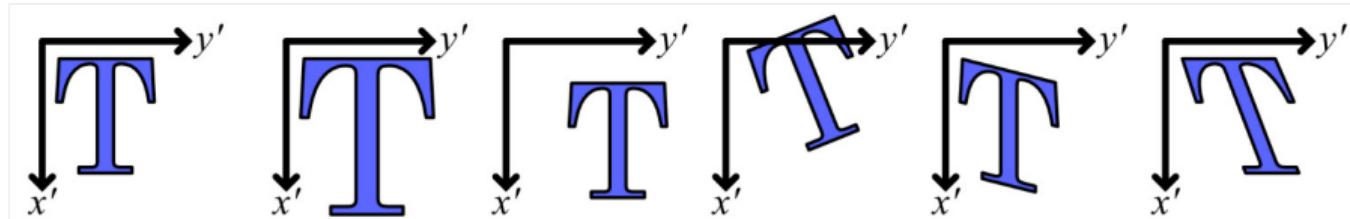
Transformations géométriques usuelles

- Sous forme d'ensemble d'équations

$$x' = a_{11}x + a_{12}y + a_{13} \quad (10)$$

$$y' = a_{21}x + a_{22}y + a_{23} \quad (11)$$

Transformations affines	Identité	Réflexion / Homothétie	Translation	Rotation	Cisaillement (vertical)	Cisaillement (horizontal)
Matrice	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 \\ s_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & s_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$



Exemples de transformations affines

Mise en œuvre pratique (1)

- **Données du problème**
- Image de départ : $f(x, y)$ sur une grille discrète
- Image transformée : $g(x', y')$ sur une grille discrète
- Transformation géométrique T
- **Nécessité d'interpoler**

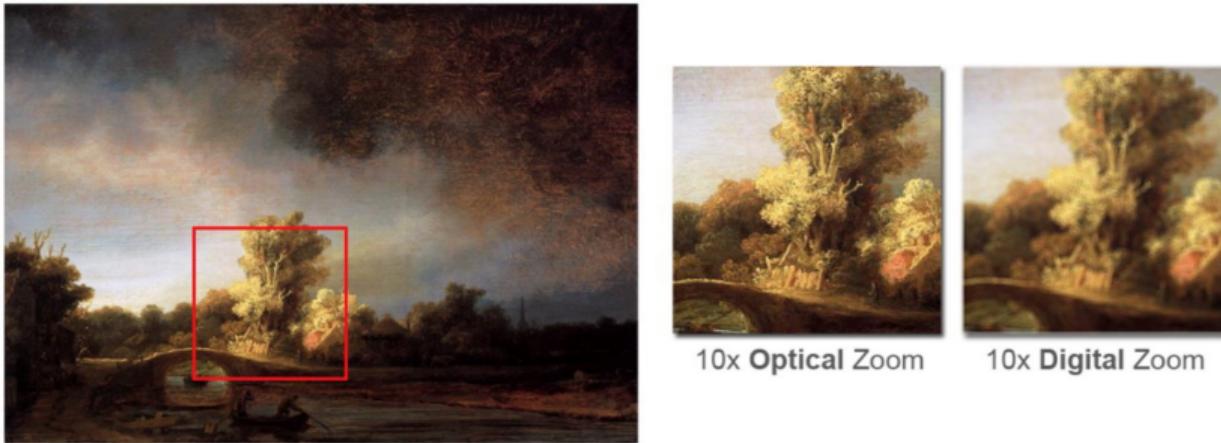
Mise en œuvre pratique (2)

- **Étapes de l'interpolation**
 - Pour chaque pixel (x', y') de l'image *transformée*
 - Déterminer les coordonnées correspondantes (x, y) dans l'image *de départ*
 - Effectuer l'interpolation dans l'image de départ
 - Affecter la valeur trouvée à l'image transformée
- Plusieurs types d'interpolation peuvent être utilisés
 - Plus proche voisin (*nearest neighbor*)
 - Bilinéaire
 - Bicubique

Interpolation d'image

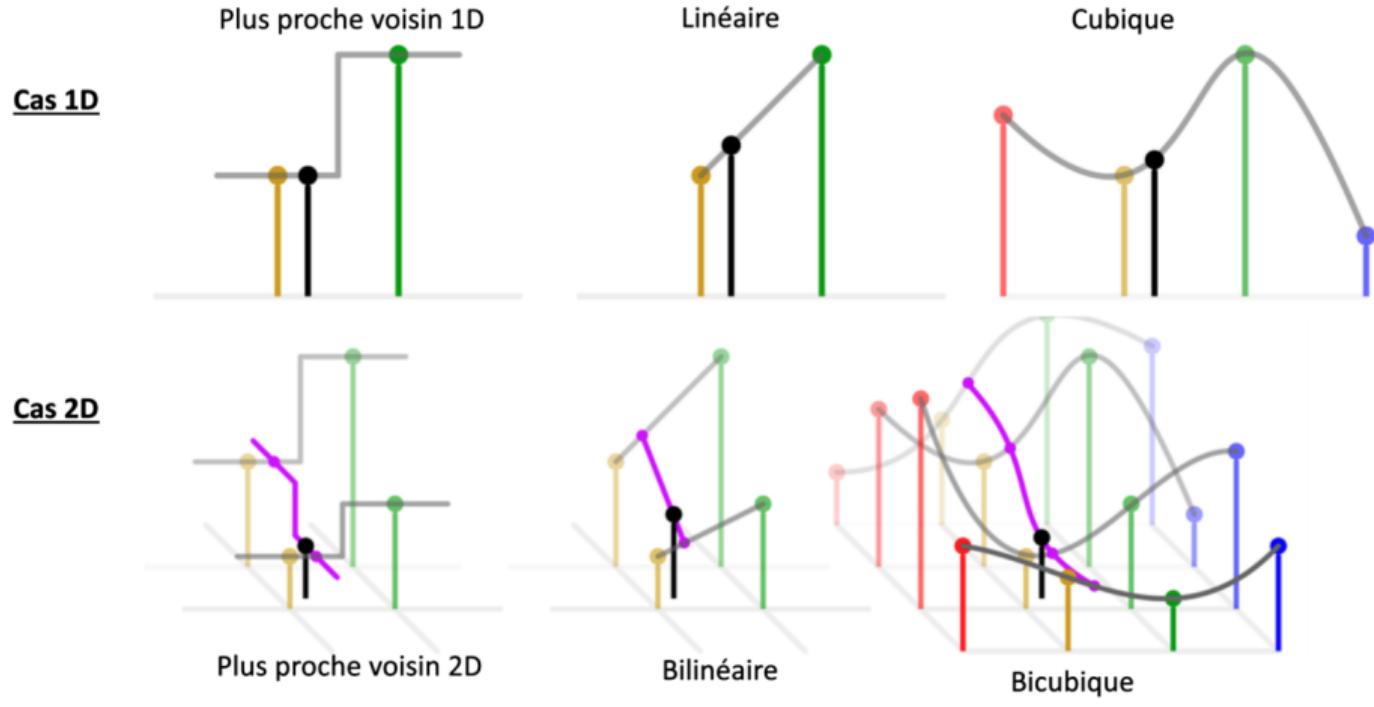
- Utilisée pour modifier la taille d'une image
- **Application** : Zoom / rétrécissement numérique
- Différence entre zooms optique et numérique ?

Example: Rembrandt's "The Stone Bridge"



Source : 2M CCTV

Plusieurs méthodes d'interpolation



[Wikipedia](#)

Exemple : Interpolation bilinéaire

- ① Interpolation selon X pour $y = y_1$

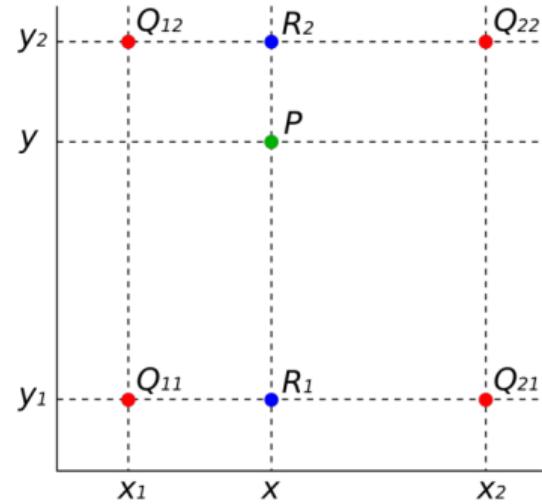
$$R_1 = \frac{x_2 - x}{x_2 - x_1} Q_{11} + \frac{x - x_1}{x_2 - x_1} Q_{21} \quad (12)$$

- ② Interpolation selon X pour $y = y_2$

$$R_2 = \frac{x_2 - x}{x_2 - x_1} Q_{12} + \frac{x - x_1}{x_2 - x_1} Q_{22} \quad (13)$$

- ③ Interpolation selon Y

$$P = \frac{y_2 - y}{y_2 - y_1} R_1 + \frac{y - y_1}{y_2 - y_1} R_2 \quad (14)$$



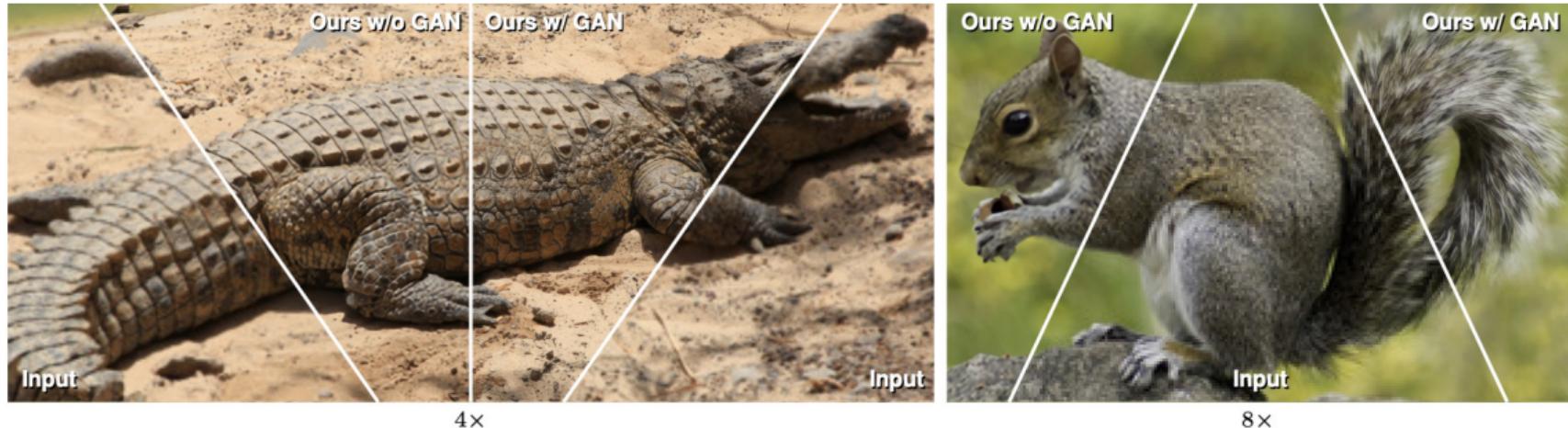
[Wikipedia](#)

Interpolation d'image et Python

- Interpolation polynomiale
- Interpolation par splines
- Utilisé par `skimage.transform.resize` et `.rescale`
- Utilisé par `plt.imshow` via l'argument `interpolation`
- **Voir la démo Jupyter**

Interpolation avancée

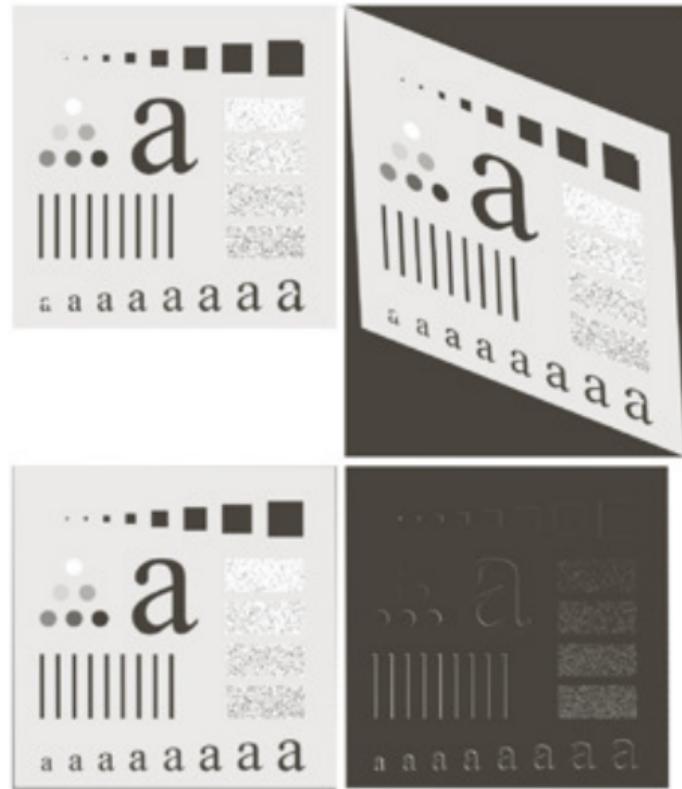
- Application IA : Super-résolution
- L'ordinateur “hallucine” les détails manquants



(Wang et al. 2018) A Fully Progressive Approach to Single-Image Super-Resolution.
([URL](#), [YouTube](#))

Application : Recalage d'images

- **Recalage** : Aligner automatiquement des images
- **But** : Estimer les transformations géométriques pour aligner 2 images à l'aide d'algorithmes.
- **Approche** : Déetecter des marqueurs communs entre les 2 images



(Gonzalez, 2018)

Section 4

Fondements du filtrage spatial

Filtrage d'image

- Calcule une fonction sur le voisinage local à toutes les positions
- Démo interactive : <https://setosa.io/ev/image-kernels/>

h = sortie

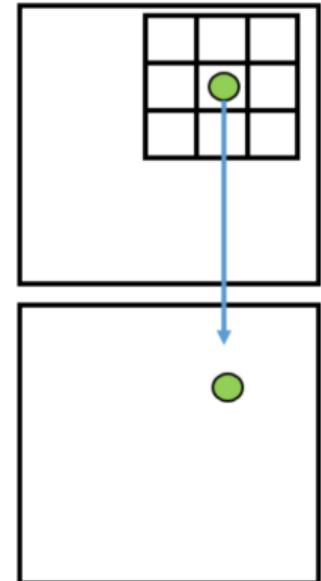
f = filtre

I = image

$$h[m, n] = \sum_{k, l} f[k, l] I[m + k, n + l]$$

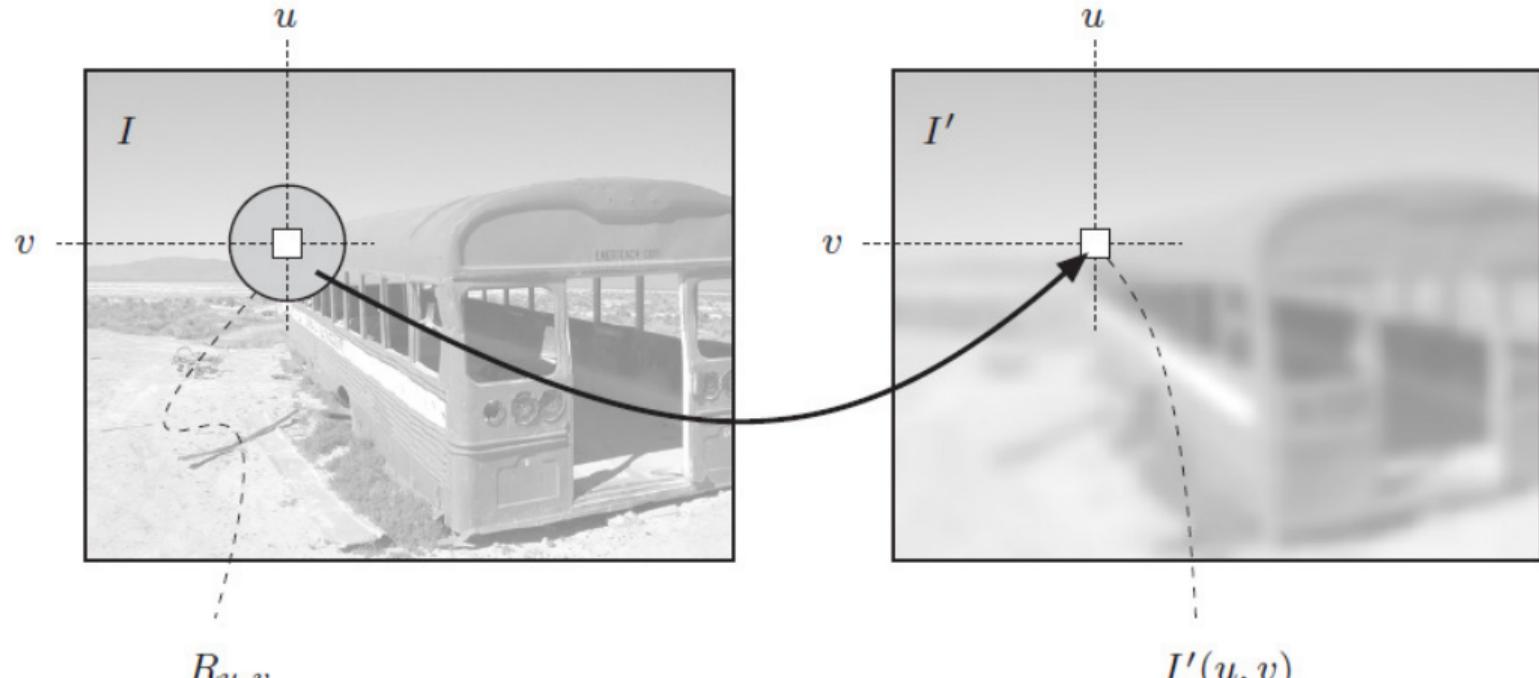
Coord. 2D = k, l

Coord. 2D = m, n



Exemple de filtre

Lien entre filtre et voisinage



(Burger2009, Vol1)

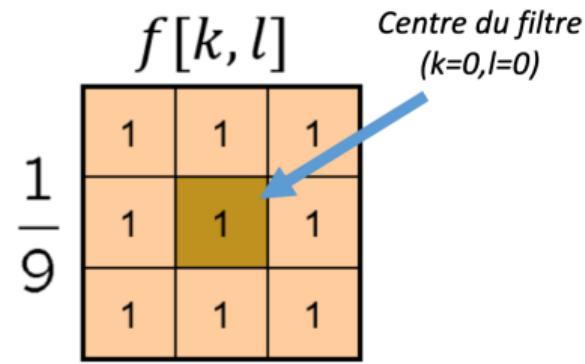
Exemple : Filtre uniforme

- Tous les éléments du noyau du filtre sont égaux
- Le filtre est **normalisé**, c.-à-d. la somme de ses éléments = 1

$$h[m, n] = \sum_{k,l} f[k, l] I[m+k, n+l] \quad (15)$$

- Filtre uniforme de taille $K \times L$

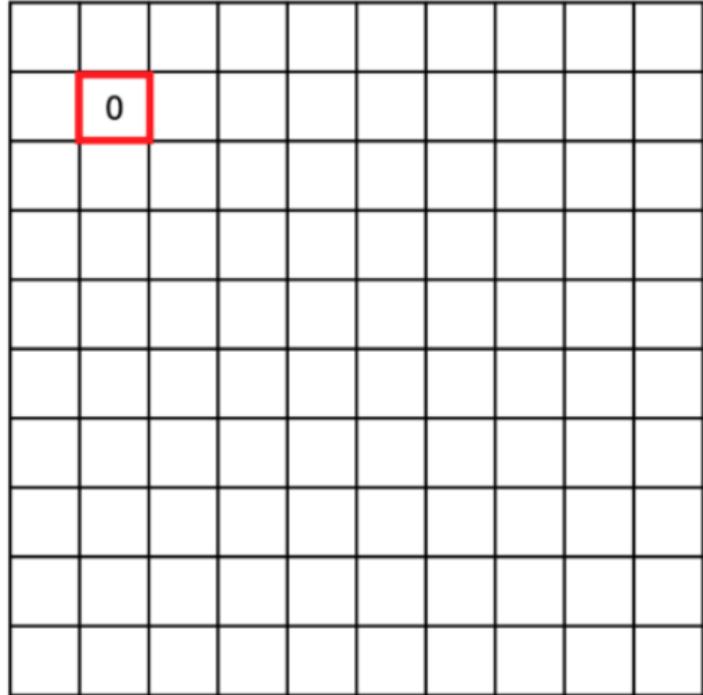
$$f[k, l] = \frac{1}{KL} \quad (16)$$



Filtre uniforme 3×3

Exemple : Filtrage d'image (1)

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



Exemple de filtrage

Exemple : Filtrage d'image (2)

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

0	10								

Exemple de filtrage

Exemple : Filtrage d'image (3)

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	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

0	10	20							

Exemple de filtrage

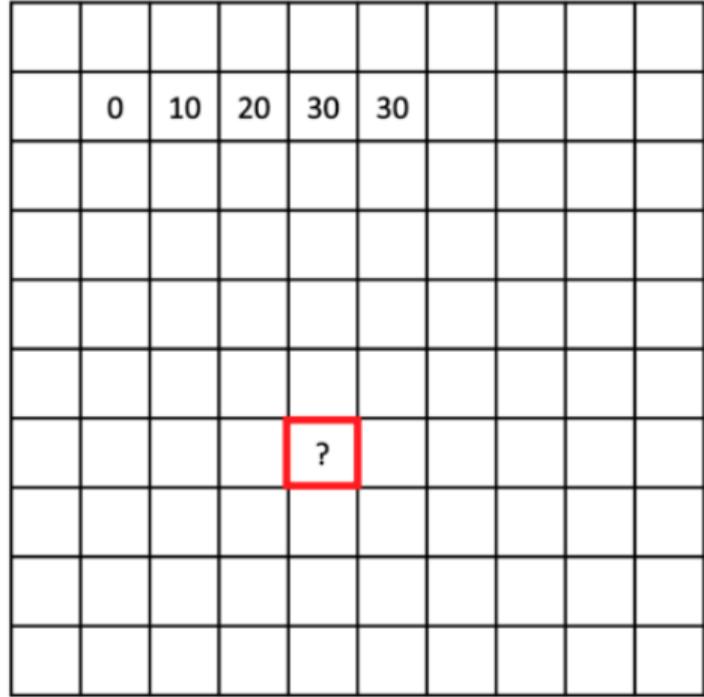
Exemple : Filtrage d'image (4)

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

0	10	20	30						

Exemple de filtrage

Exemple : Filtrage d'image (5)



Exemple de filtrage

Exemple : Filtrage d'image (6)

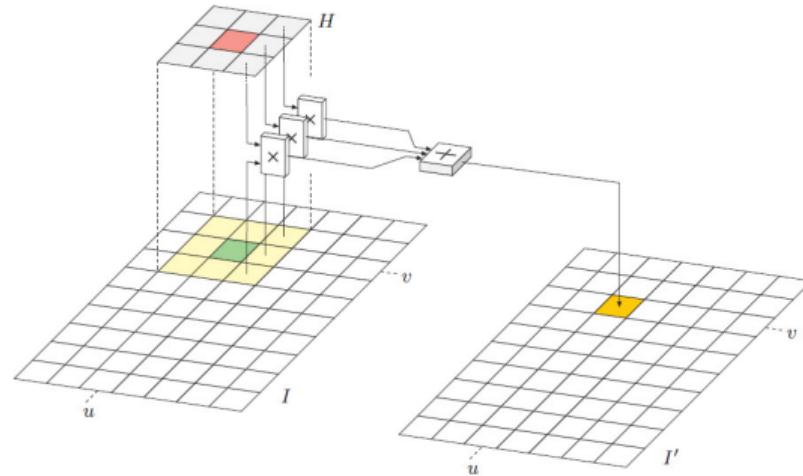
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	0	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

	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		

Exemple de filtrage

Retour sur le filtre uniforme

- **Qu'est-ce qu'il accomplit ?**
- Remplace chaque pixel par la moyenne de son voisinage.
- Ce filtre a un effet lissant (retire les détails fins).
- **Pourquoi est-ce que la somme = 1 ?**



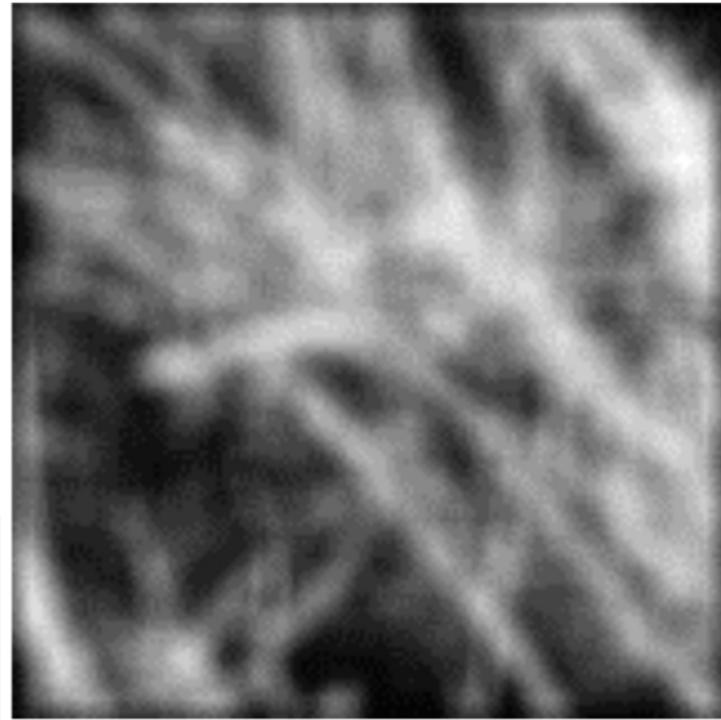
(Burger2009, Vol1)

Exemple : Lissage avec un filtre uniforme



$f[k, l]$

1	1	1
1	1	1
1	1	1



Effet du filtre uniforme

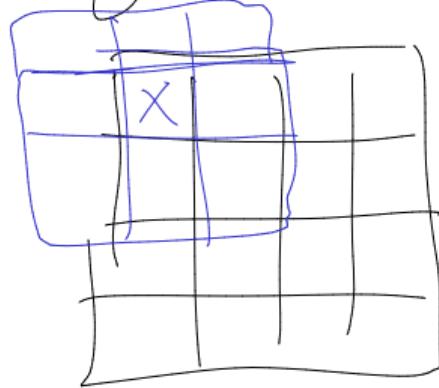
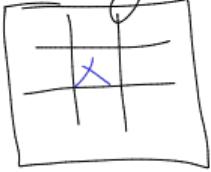
Filtrage d'image

- **Filtrage d'image** : Calcule une fonction sur le voisinage local à toutes les positions

$$h[m, n] = \sum_{k,l} f[k, l]I[m + k, n + l] \quad (17)$$

- Opération très importante !
- **Rehausse les images** (ex. : débruitage, changement d'échelle, rehaussement du contraste)
- **Extraction d'information contenue dans l'image** (ex. : texture, contours, points distincts)
- **Détection de motifs** (ex. *template matching*)

noyau (filtrage)



Question : Effet des filtres



1.

0	0	0
0	1	0
0	0	0

2.

0	0	0
0	0	1
0	0	0

decaler
vers gauche

3.

1	0	-1
2	0	-2
1	0	-1

4.

0	0	0
0	2	0
0	0	0

-

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

Sobel \rightarrow

Quelques filtres

Effet du filtre 1

$$h[m, n] = \sum_{k,l} f[k, l]I[m + k, n + l] \quad (18)$$



Original

0	0	0
0	1	0
0	0	0

Filtre 1



Filtré
(pas de changement)

Effet du filtre 2

$$h[m, n] = \sum_{k,l} f[k, l]I[m + k, n + l] \quad (19)$$



0	0	0
0	0	1
0	0	0

Original

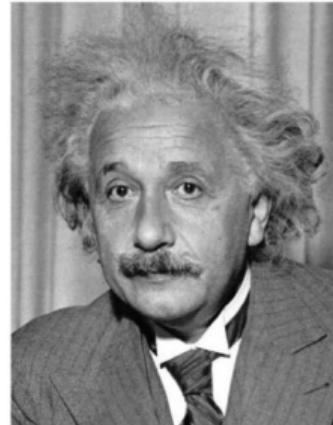


Décalé vers la gauche
de 1 pixel

Filtre 2

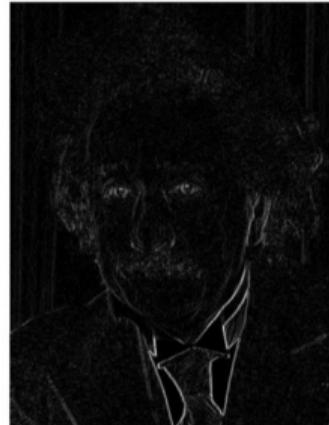
Effet du filtre 3

$$h[m, n] = \sum_{k,l} f[k, l]I[m + k, n + l] \quad (20)$$



1	0	-1
2	0	-2
1	0	-1

Sobel

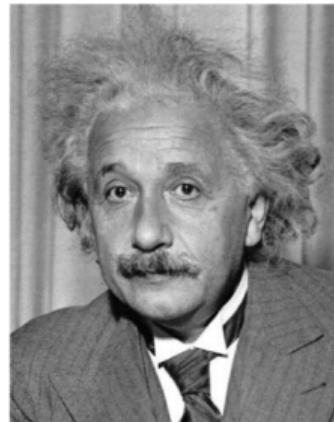


Contours **verticaux**
(valeur absolue)

Filtre 3

Effet du filtre 3 transposé

$$h[m, n] = \sum_{k,l} f[k, l]I[m + k, n + l] \quad (21)$$



1	2	1
0	0	0
-1	-2	-1

Sobel



Contours horizontaux
(valeur absolue)

Filtre 3 transposé

Effet du filtre 4

$$h[m, n] = \sum_{k,l} f[k, l]I[m + k, n + l] \quad (22)$$



$$\begin{matrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{matrix} - \frac{1}{9} \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix}$$

(Notez que la somme des éléments du filtre = 1)

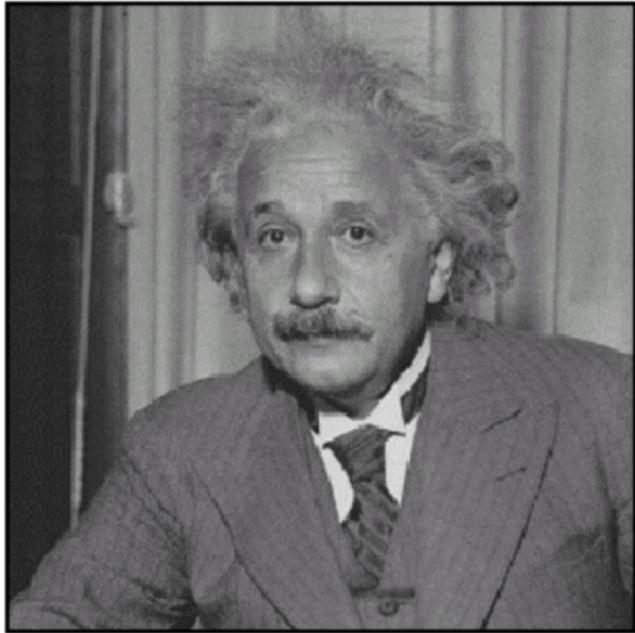
Original

Filtre 4

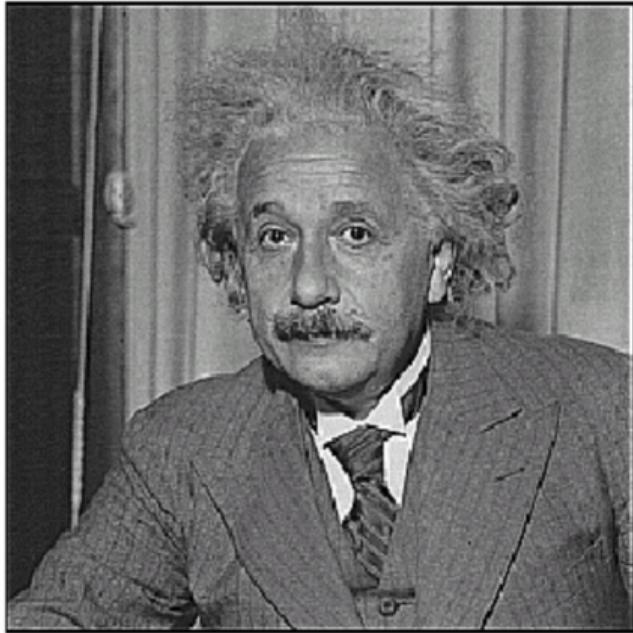


Filtre rehausseur (sharpening filter).
Accentue les différences avec la moyenne locale

Effet du filtre rehausseur



Avant



Après

Filtre rehausseur

Corrélation et convolution

- **Corrélation 2D**

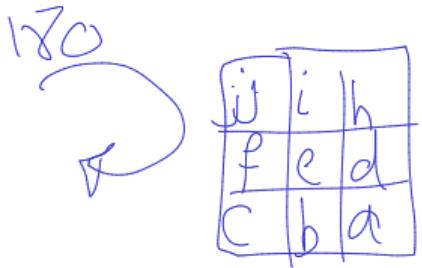
$$h[m, n] = \sum_{k,l} f[k, l]I[m + k, n + l] \quad (23)$$

- **Exemple** : `h = scipy.signal.correlate2d(f, I)`
- **Convolution 2D**

$$h[m, n] = \sum_{k,l} f[k, l]I[m - k, n - l] \quad (24)$$

- **Exemple** : `h = scipy.signal.convolve2d(f, I)`
- **Note** : Une convolution est similaire à une corrélation, mais avec un filtre pivoté de 180°. La corrélation et la convolution sont identiques lorsque le noyau du filtre est symétrique.

a	b	c
d	e	f
h	i	j



Propriétés importantes des filtres linéaires

- **Linéarité**
 - $\text{imfilter}(I, f_1 + f_2) = \text{imfilter}(I, f_1) + \text{imfilter}(I, f_2)$
- **Invariance à la translation**
 - Même résultat pour même intensité, peu importe la position du pixel (m, n)
 - $\text{imfilter}(I, \text{shift}(f)) = \text{shift}(\text{imfilter}(I, f))$
- Tout filtre linéaire et invariant à la translation peut être représenté par une convolution.

↳ si on applique filtre en haut ou en bas
ca donne même résultat

Propriétés des convolutions

- **Commutativité : $a * b = b * a$**
 - Conceptuellement, il n'y a pas de différence entre le filtre et le signal.
 - Mais, pour certaines implémentations, la commutativité n'est pas respectée, p. ex. Bord de l'image
- **Associativité : $a * (b * c) = (a * b) * c$**
 - On peut souvent utiliser plusieurs filtres un après l'autre
 - **Exemple :** $((a * b1) * b2) * b3$
 - Ceci est équivalent à appliquer un seul filtre : $a * (b1 * b2 * b3)$
 - Une corrélation n'est pas associative (effet de rotation)

Autres propriétés des convolutions

- **Distributivité de l'addition**

$$a * (b + c) = (a * b) + (a * c) \quad (25)$$

- **Produit par un scalaire**

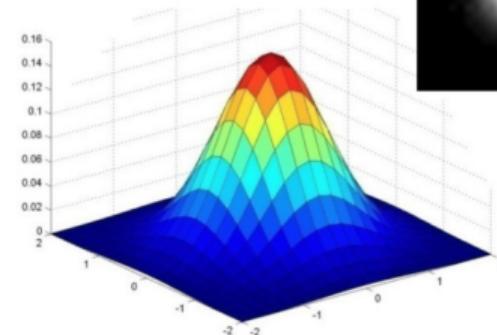
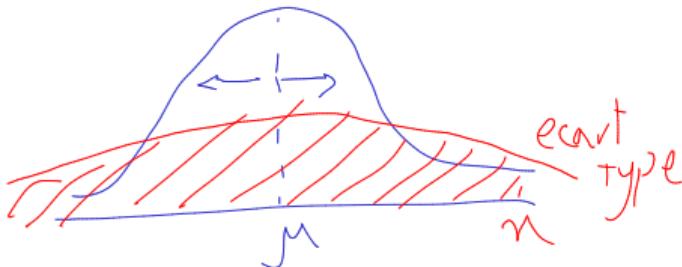
$$ka * b = a * kb = k(a * b) \quad (26)$$

- **Identité** (*réponse unitaire* $e = [0, 0, 1, 0, 0]$)

$$a * e = a \quad (27)$$

Filtre important : Gaussien

- Pondère la contribution des pixels voisins par leur proximité.



$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$G(x) = e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (28)$$

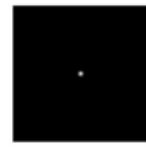
x	0.003	0.013	0.022	0.013	0.003
y	0.013	0.059	0.097	0.059	0.013
x	0.022	0.097	0.159	0.097	0.022
y	0.013	0.059	0.097	0.059	0.013
x	0.003	0.013	0.022	0.013	0.003

$5 \times 5, \sigma = 1$

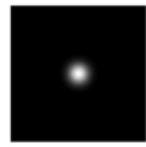
$$\rightarrow \sum G_{ij} = 1$$

Filtre gaussien 5×5

Filtres gaussiens



$\sigma = 1$ pixel



$\sigma = 5$ pixels



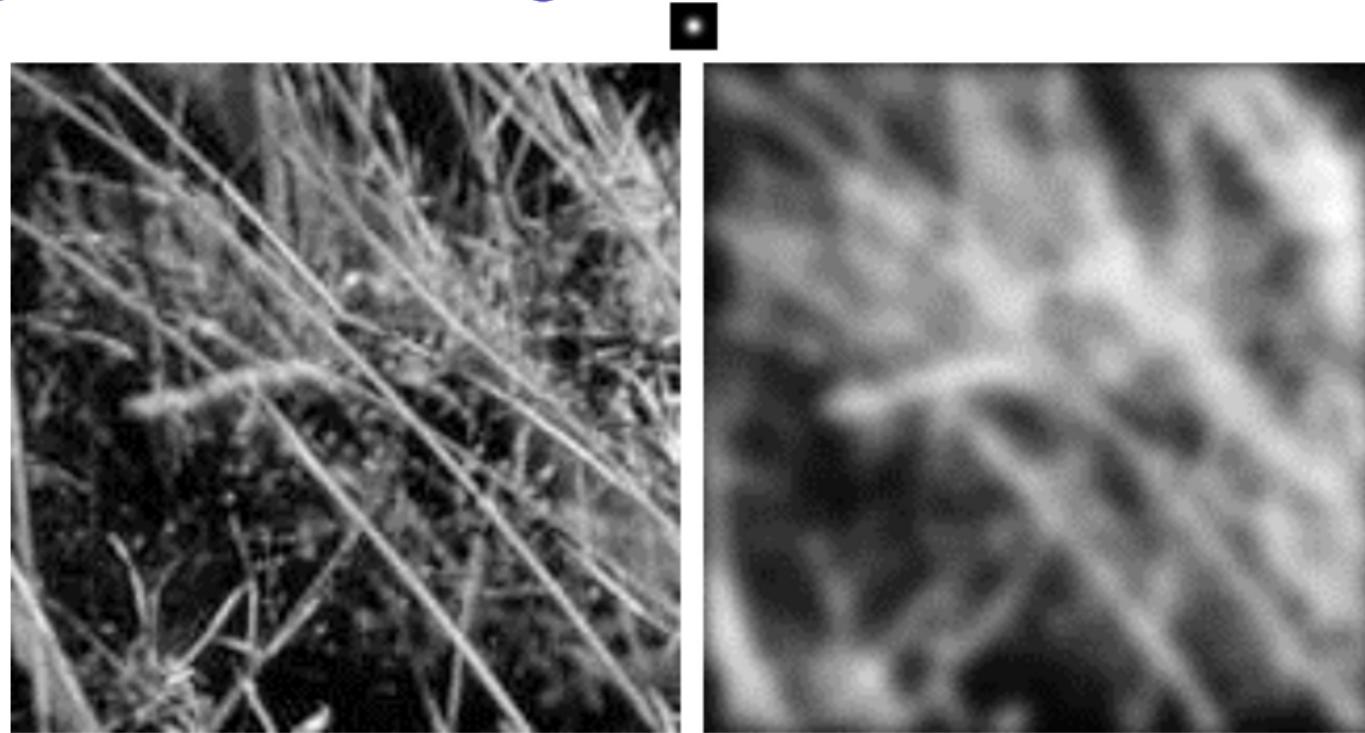
$\sigma = 10$ pixels



$\sigma = 30$ pixels

Filtres gaussiens

Lissage avec un filtre gaussien

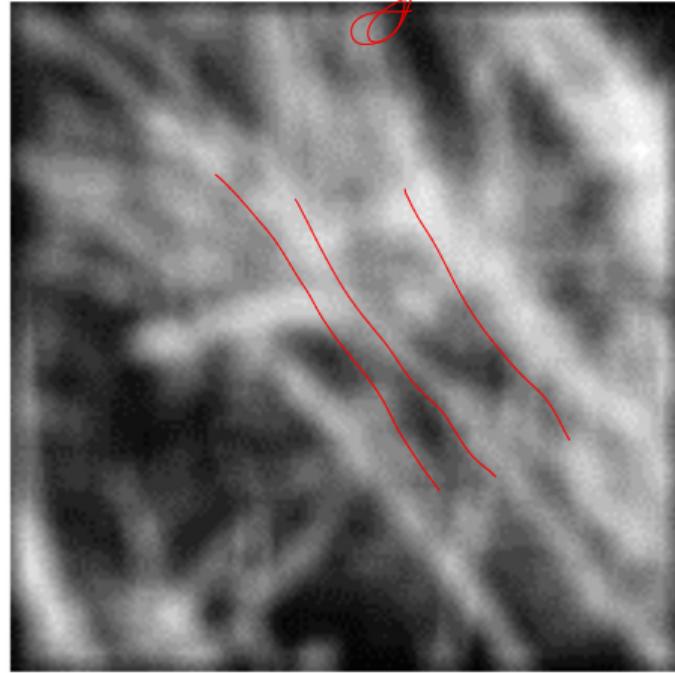


Filtre gaussien

Lissage avec un filtre uniforme



on voit les ligne sur diag
mais au
G(1) pas
de dif



Filtre uniforme

Information du filtre gaussien

$G \times G \rightarrow G^{(1)}$ plus efficac
 $G_1 * G_2 \approx G_{1+2}$

- Retire les **hautes fréquences** dans l'image (*filtre passe-bas*)
- **Conséquence** : Les images deviennent plus **lisses**
- Une gaussienne convoluée avec une gaussienne est ... une autre gaussienne !
- On peut donc lisser une image avec une gaussienne de faible diamètre, répéter, et obtenir le même résultat qu'un lissage avec une gaussienne équivalente de plus grand diamètre.
- Deux convolutions consécutives avec un noyau gaussien de diamètre σ est équivalent à convoluer 1 seule fois par une gaussienne de diamètre $\sigma\sqrt{2}$
- **Noyau séparable** : Factorisation en un produit de 2 gaussiennes 1D

Séparabilité du filtre Gaussien

- Une fonction gaussienne 2D $G_\sigma(x, y)$ peut être exprimée en tant que produit de 2 fonctions 1D $f(x)g(y)$
- Dans ce cas, les deux fonctions sont une gaussienne 1D (identique)

$$G_\sigma(x, y) = \frac{1}{2\pi\sigma^2} \exp -\frac{x^2 + y^2}{2\sigma^2} \quad (29)$$

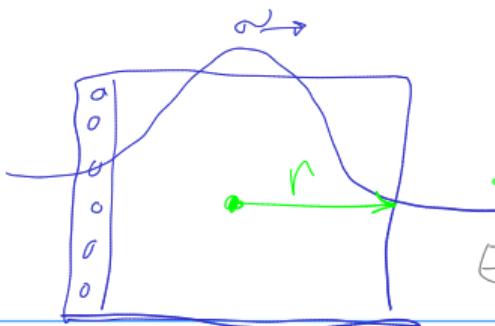
$$= \left(\frac{1}{\sqrt{2\pi}\sigma} \exp -\frac{x^2}{2\sigma^2} \right) \left(\frac{1}{\sqrt{2\pi}\sigma} \exp -\frac{y^2}{2\sigma^2} \right) \quad (30)$$

Séparabilité

- **Pourquoi est-ce que la séparabilité est utile ?**
- Efficacité computationnelle
- **Exemples**
 - **Image de taille** : $M \times N$
 - **Filtre de taille** : $P \times Q$
 - **Convolution 2D** : $MNPQ$ (Additions / Multiplications)
 - **Séparable 2D** : $MN(P + Q)$ (Additions / Multiplications)
 - **Accélération** : $PQ/(P + Q)$
 - Pour un filtre $9 \times 9 = 4.5x$ plus rapide

Détails pratiques : taille d'un filtre Gaussien

- **Quelle devrait-être la taille d'un filtre Gaussien ?**
- Les valeurs près du **bord** devraient être **proches de zéro**.
- Les filtres gaussiens ont un **support infini** (la fonction est définie partout dans l'espace)
- **Règle de base pour la Gaussienne** : choisir un **rayon** de filtre r entre 2.5σ et 3.5σ (typiquement $r = 3\sigma$).
- La **taille** du filtre $N \times N$ est alors $N = 2r + 1$ (on souhaite un filtre de taille impaire)
- **Exemple** : Pour $G_{\sigma=1}$, la taille idéale du filtre est de 7×7
- Voir ([Burger2009, Vol1, Section 5.3.3](#)) pour plus de détails.



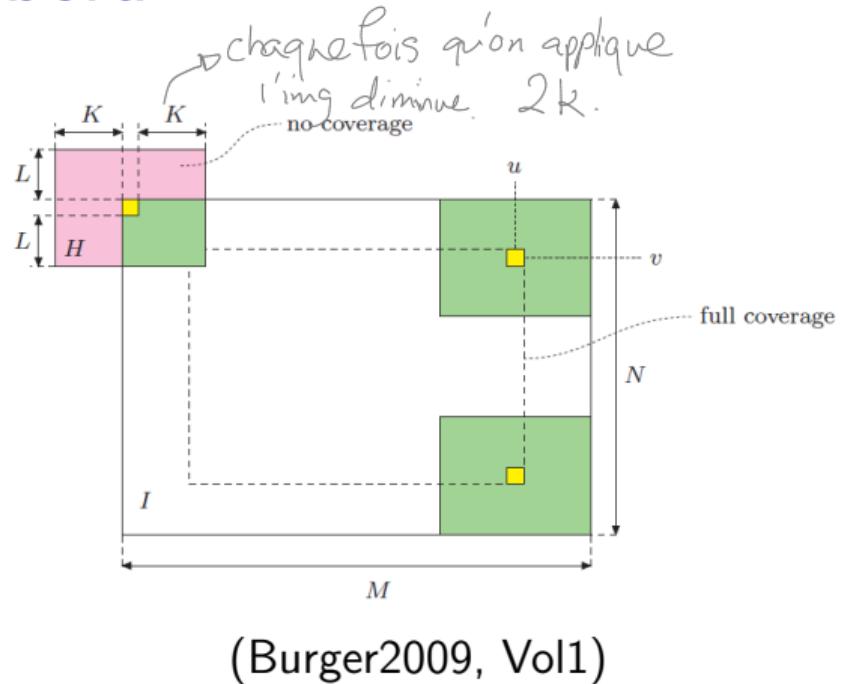
dont être = $r = 2.5\varphi - 3.5\varphi$

Ex: $\varphi = 1 \Rightarrow 3 \rightarrow 3 \times 2 + 1$

7

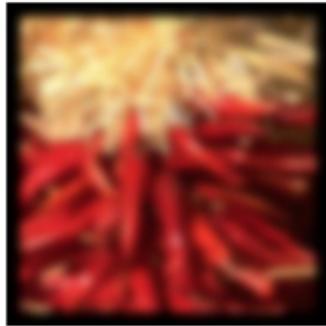
Détails pratiques : Effet de bord

- Qu'arrive-t-il près du bord de l'image ?
- Une partie du voisinage du filtre est à l'extérieur de l'image.
- Nécessite une **extrapolation**
- Méthodes :
 - Valeur continue
 - Périodique
 - Copie du bord
 - Réflexion (miroir)



Méthodes d'extrapolation (padding)

Valeur continue



Périodique



Copie du bord

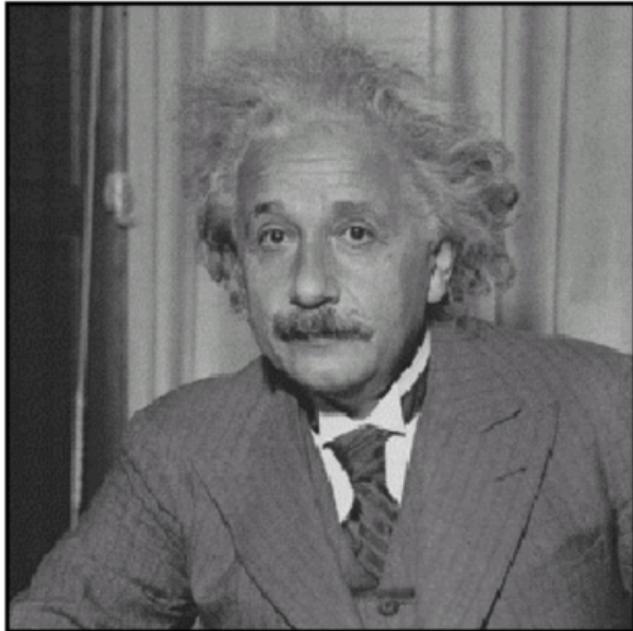


Réflexion (Miroir)

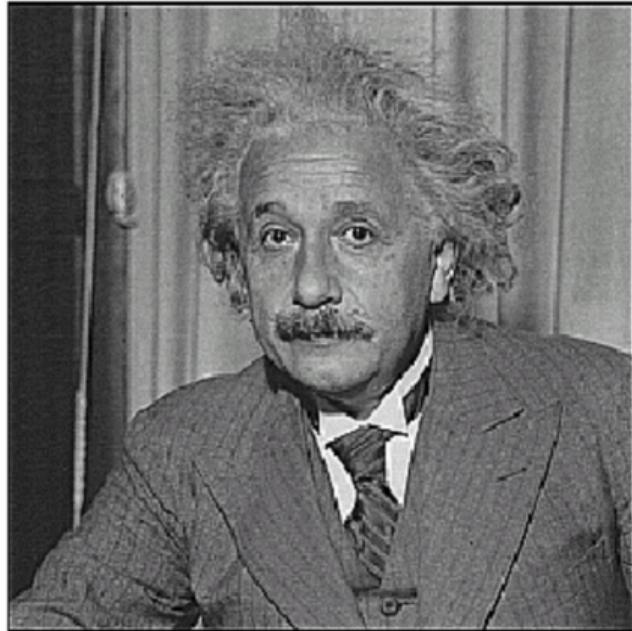


Exemples d'extrapolation

Application : Affinage (*Sharpening*)



Avant



Après

Filtre rehausseur

Principe de l'affinage (1)

- **Qu'est-ce que le lissage retire dans l'image originale ?**
- Les détails fins (*haute fréquence*)

pour donner cette résultat
il faut créer, après filtrer
l'original

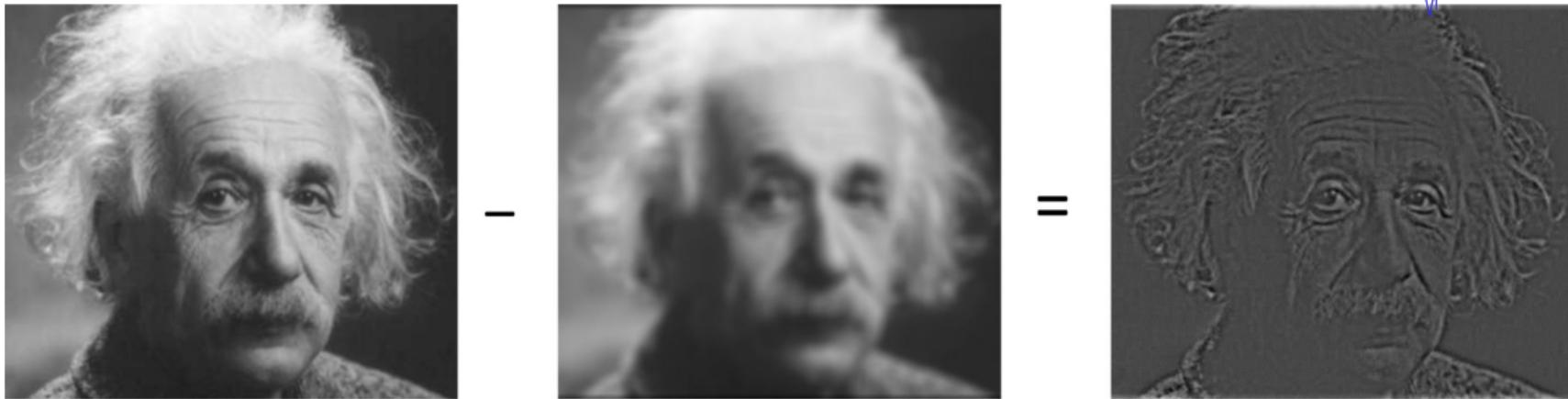


Image originale (gauche), image lissée (centre) et différence entre les deux images.

Principe de l'affinage (2)

- Le but de l'affinage est donc d'amplifier les détails fins dans l'image originale.

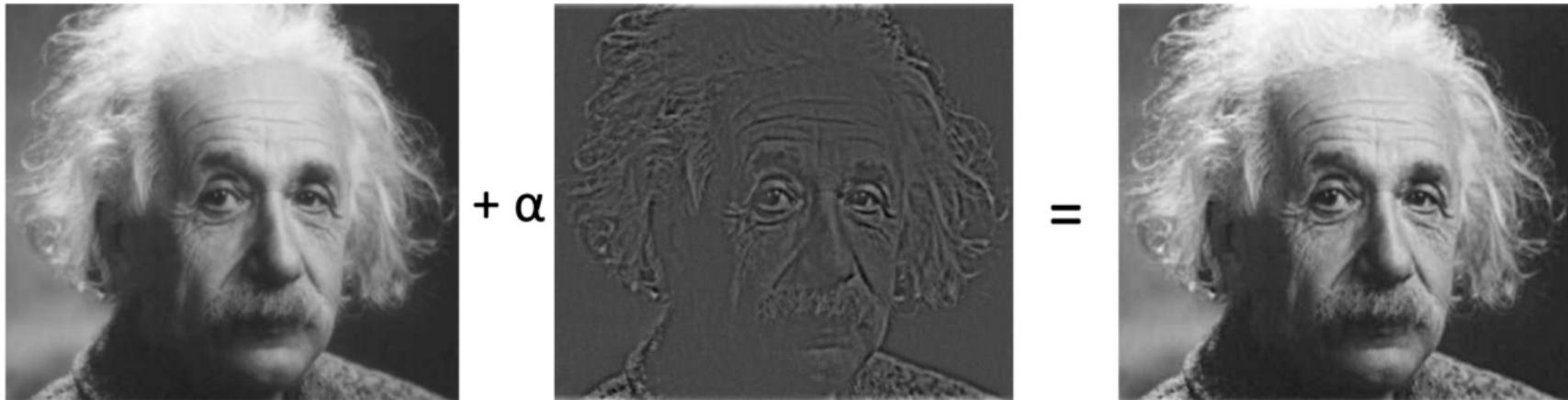
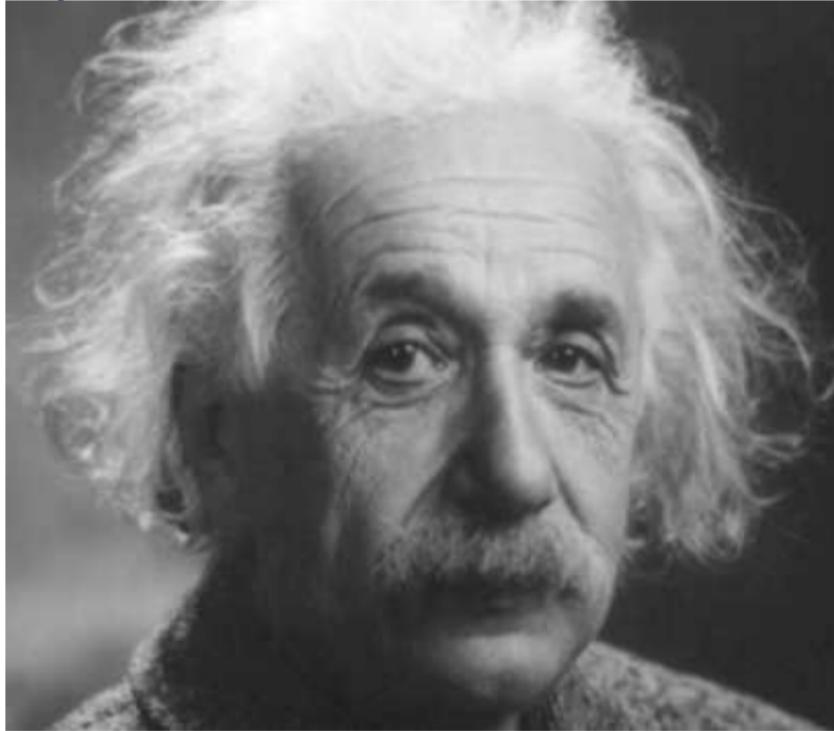


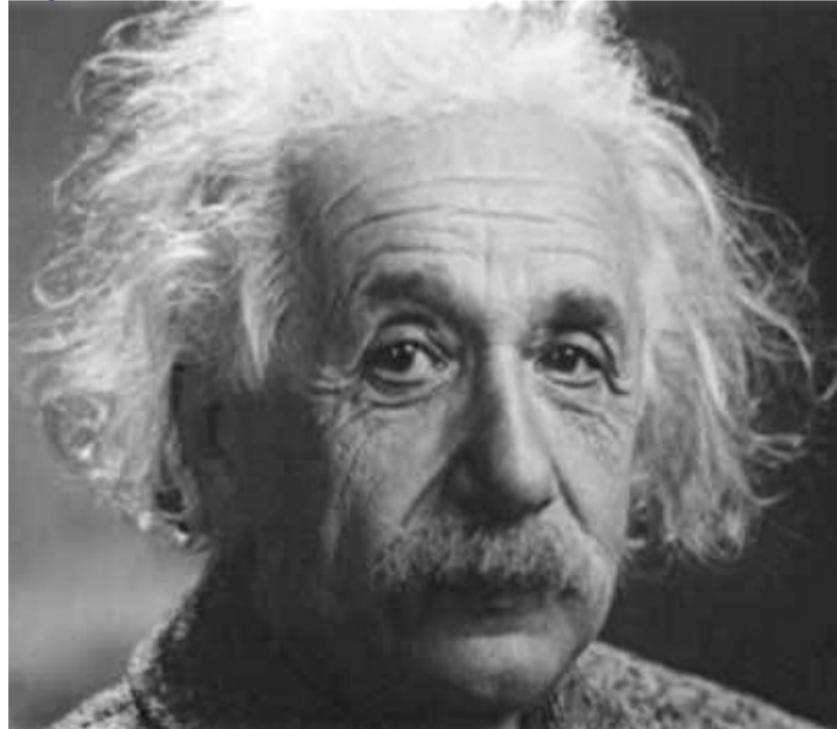
Image originale (gauche), hautes fréquences (centre), et image affinée (droite)

Affinage (avant)



Avant l'affinage

Affinage (après)



Après l'affinage

Exemple : Affinage avec un seul filtre

$\sim \text{intensity}_\text{frou}$

$$f' = f + \alpha(f - f_{\text{frou}})$$

$\sim \text{intensity}_\text{fou}$

$$f' = (1 + \alpha)f - \alpha f_{\text{frou}}$$

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

$$f' = (1 + \alpha)(w * f) - \alpha(v * f)$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$f' = ((1 + \alpha)w - \alpha v) * f$$

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{pmatrix} - \frac{1}{9} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

~~unsharp filter~~
~~unsharp-mask~~
(scikit-image)



Original

$$* \left(\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{array} - \frac{1}{9} \begin{array}{ccc} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{array} \right)$$



Filtre rehausseur
(Accentue les
contours)

Exemple : Masque flou (affinage)

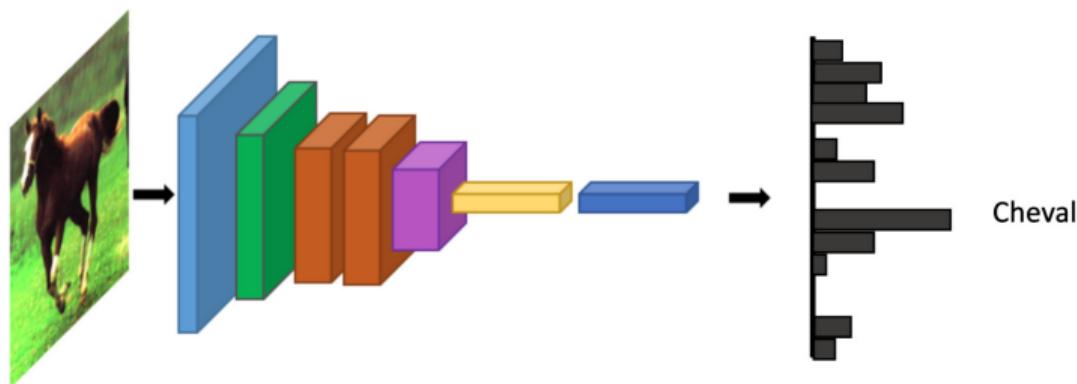
- ① Lissage de l'image originale
- ② Soustraction de l'image originale par l'image floue
- ③ Addition du masque flou (*unsharp*) et de l'image originale



(Gonzalez, 2018)

Convolution et *Convolutional Neural Networks* (CNNs)

- La convolution est l'opération de base des CNNs
- Les noyaux des convolutions sont **appris**, ce qui permet d'obtenir des caractéristiques utiles pour décrire les informations contenues dans les images



Classification avec un CNN

Convolution “optique”



Instabilité d'un caméra. Source: Fergus, et al. "Removing Camera Shake from a Single Photograph", SIGGRAPH 2006



Bokeh : Flou des régions hors focus de l'image. ([URL](#))

Section 5

Filtres non-linéaires

Filtres médian / min / max ...

- Agit sur une fenêtre en choisissant l'intensité médiane dans cette fenêtre.
- Principes des filtres de classement (*rank filter*)
- **Exemple** : filtre min, max, étendue, médiane, percentile

1	2	0	0	0
1	3	1	0	1
0	1	2	3	2
0	1	0	2	3
1	1	1	2	3

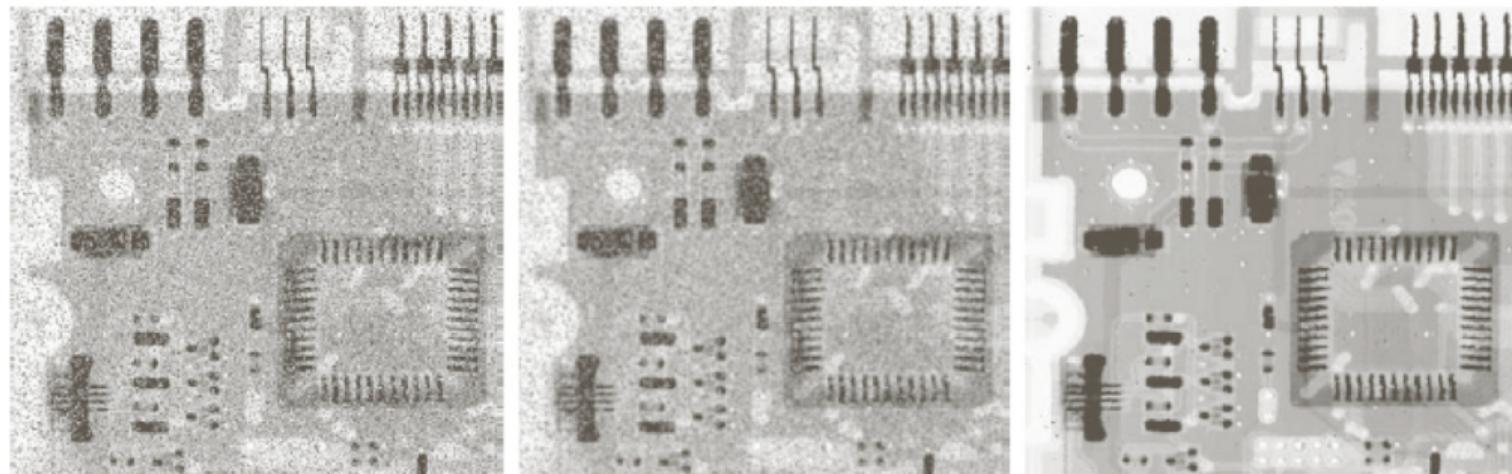


- Classement des intensités du voisinage 5x5
[0,0,0,0,0,0,1,1,1,1,1,1,1,1,2,2,2,2,3,3,3]
- Sélection d'une valeur basée sur le rang dans ce classement
- Exemples : min = 0, max=3, médiane = 1, ...

Filtre de classement

Exemple : Adoucissement (*smoothing*)

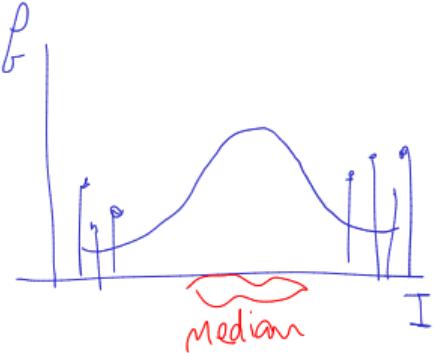
- **Filtre médian**
- Efficace pour certains types de bruits (ex. : impulsif)
- Généralisations : filtres à percentile



(Gonzalez, 2018)

Exemple : Bruit de type Sel et poivre

Image bruitée (Type Sel et Poivre)



Bruit de type *sel et poivre*

Exemple : Filtre uniforme 3×3

Image bruitée (Type Sel et Poivre)



Filtre uniforme 3×3



Filtre uniforme 3×3

Exemple : Filtre uniforme 11×11

Image bruitée (Type Sel et Poivre)



Filtre uniforme 11×11



Filtre uniforme 11×11

Exemple : Filtre médian 3×3

Image bruitée (Type Sel et Poivre)



Filtre médian 3×3



Filtre uniforme 3×3

Exemple : Filtre médian 11×11

Image bruitée (Type Sel et Poivre)



Filtre médian 11×11



Filtre uniforme 11×11

Comparaison entre les filtres

- Chaque filtre a une application différente
- Les paramètres du filtre influencent les résultats
- Nous verrons plusieurs autres filtres au cours de la session (ex. : détection de contours)



Comparaison

Section 6

Démo Python

Démo Python

- Module skimage.exposure
- Module scipy.ndimage.filters
- Module skimage.filters
- Histogrammes avec numpy et matplotlib