ESIEE PARIS

Traitement et analyse d'images

Benjamin Perret

une école de la Chambre de commerce et d'industrie de Paris

Filtre

1. Introduction
    a) Exemple : filtre moyenneur
    b) Filtre global, semi-local, local et adaptatif

2. Filtre linéaire - Convolution
    a) Formulation
    b) Smoothing
    c) Sharpening
    d) Implémentation

3. Filtre non linéaire
    a) Filtre de rang, filtre médian
    b) Morphologie binaire
    c) Morphologie en niveau de gris

4. Binarisation

Filtre

1. Introduction
   a) Exemple : filtre moyenneur
   b) Filtre global, semi-local, local et adaptatif

2. Filtre linéaire - Convolution
   a) Formulation
   b) Smoothing
   c) Sharpening
   d) Implémentation

3. Filtre non linéaire
   a) Filtre de rang, filtre médian
   b) Morphologie binaire
   c) Morphologie en niveau de gris

4. Binarisation

Filtre
Introduction

- But du filtrage :
  - obtenir une nouvelle image à partir d'une image

- Améliorer la qualité
  - esthétique
  - scientifique

Filtre
Exemple – Le filtre moyenneur

- Principe :
  - La valeur filtrée du pixel *p* est égale à la moyenne des valeurs des pixels voisins de *p*

- Effet : on va lisser l'image
  - réduire le bruit
  - flouter les contours et les textures

Filtre
Exemple – Le filtre moyenneur

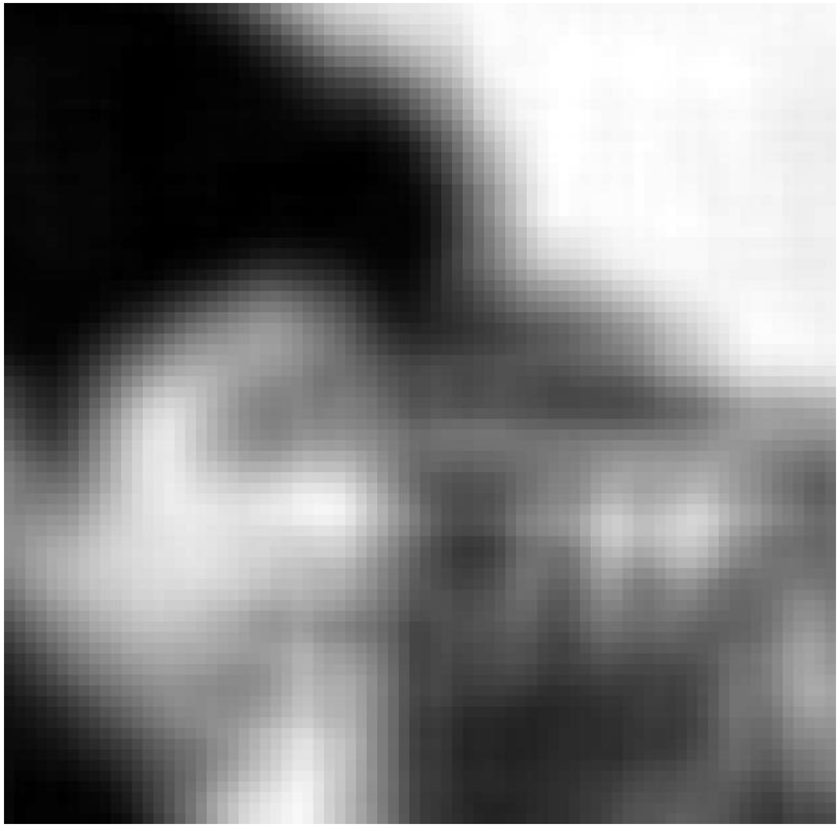
Originale

Filtrée

Filtre

Exemple – Le filtre moyenneur

Originale



Filtrée

Filtre

Exemple – Le filtre moyenneur

- Détail de calcul
  - Pour une fenêtre de 3*3 autour du pixel

24	32	128	240	255
12	42	111	154	222
4	23	123	176	243
15	63	145	134	172
27	12	98	75	143

→

		108		

$$108 = \frac{42+111+154+23+123+176+63+145+134}{9}$$

Filtre

Exemple – Le filtre moyenneur

- Détail de calcul
  - Pour une fenêtre de 5*5 autour du pixel

24	32	128	240	255
12	42	111	154	222
4	23	123	176	243
15	63	145	134	172
27	12	98	75	143

		107		

$$107 = \frac{24+32+128+240+255+12+\ldots+98+75+143}{25}$$

Filtre

Exercice – Complexité du filtre moyenneur

- On considère une image de n*n pixels
- On prend un voisinage de m*m pixels

- Combien d’opérations (addition, multiplication) sont nécessaires pour le calcul du filtre moyenneur ?

Filtre

Exemple – Le filtre moyenneur

- Remarques:
  - La valeur d'un pixel ne dépend que des pixels proches : on parle d'approche locale
  - Le voisinage des pixels sont tous identiques (à une translation prêt)

Filtrage
Filtre global

- Lorsque la valeur d'un pixel dépend de la valeur de tous les autres pixels on parle de filtre global

- La modification d'un seul pixel de l'image originale peut impacter la valeur de tous les pixels du résultat

Filtrage
Filtre local

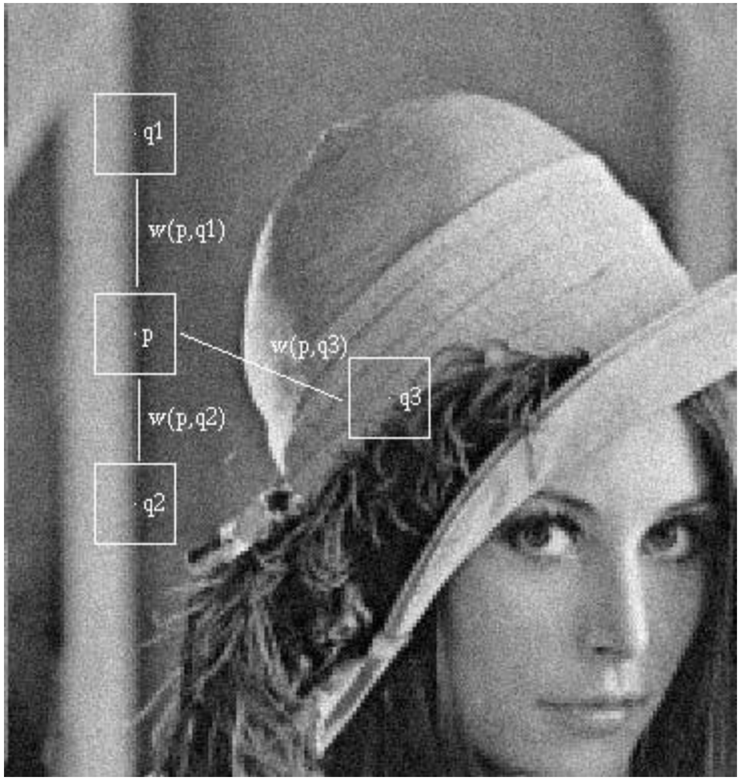
- Lorsque la valeur d'un pixel dépend de la valeur des pixels dans son voisinage on parle de filtre local

- La modification d'un seul pixel de l'image originale ne peut impacter que la valeur des pixels proches dans le résultat

Filtrage
Filtre semi-local

- Lorsque la valeur d'un pixel dépend de la valeur des pixels dans un voisinage étendu on parle de filtre semi-local

- Idée : utiliser des pixels ayant un voisinage similaire au pixel courant
  – Préserver les contours, les textures

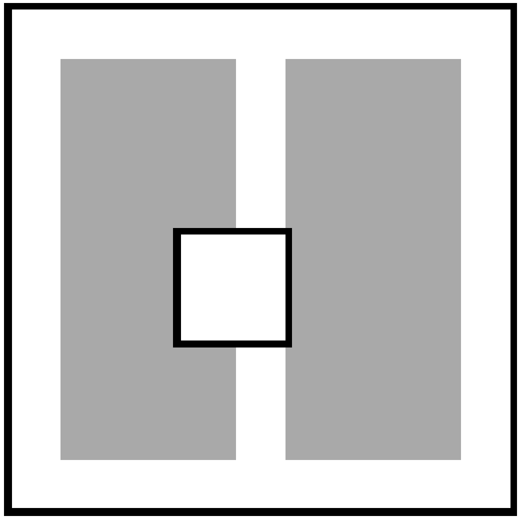


Filtrage

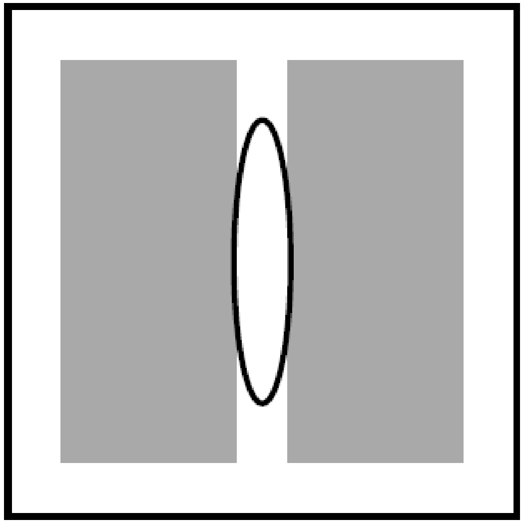
Filtre adaptatif

- Prendre un voisinage carré n'est pas toujours idéal
- Si le voisinage utilisé change en fonction de la position de l'image on parle de filtre adaptatif

- Idée : préserver les contours,



non adaptatif



adaptatif

Filtre

1. Introduction
    a) Exemple : filtre moyenneur
    b) Filtre global, semi-local, local et adaptatif

2. Filtre linéaire - Convolution
    a) Formulation
    b) Smoothing
    c) Sharpening
    d) Implémentation

3. Filtre non linéaire
    a) Filtre de rang, filtre médian
    b) Morphologie binaire
    c) Morphologie en niveau de gris

4. Binarisation

Filtre linéaire

- On appelle filtre linéaire toute fonction

$$\sigma : E^D \mapsto E^D$$

qui

- à partir d'une image renvoie une nouvelle image
- respecte les propriétés de linéarité

$$\forall f, g \in E^D,\ \sigma(f+g) = \sigma(f) + \sigma(g)$$

$$\forall f \in E^D,\ \forall k \in \mathbb{R},\ \sigma(kf) = k\sigma(f)$$

Filtre linéaire
Exercice

- Montrer que le filtre moyenneur est un filtre linéaire.

Convolution

- Soient 2 images $f, w : \mathbb{Z}^2 \mapsto \mathbb{R}$
- On note le produit de convolution de *f* par *w* :

$$f * w$$

- Définit par : $\forall (x, y) \in \mathbb{Z}^2$

$$(f * w)(x, y) = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} f(x - i, y - j) w(i, j)$$

$$= \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} f(i, j) w(x - i, y - j)$$

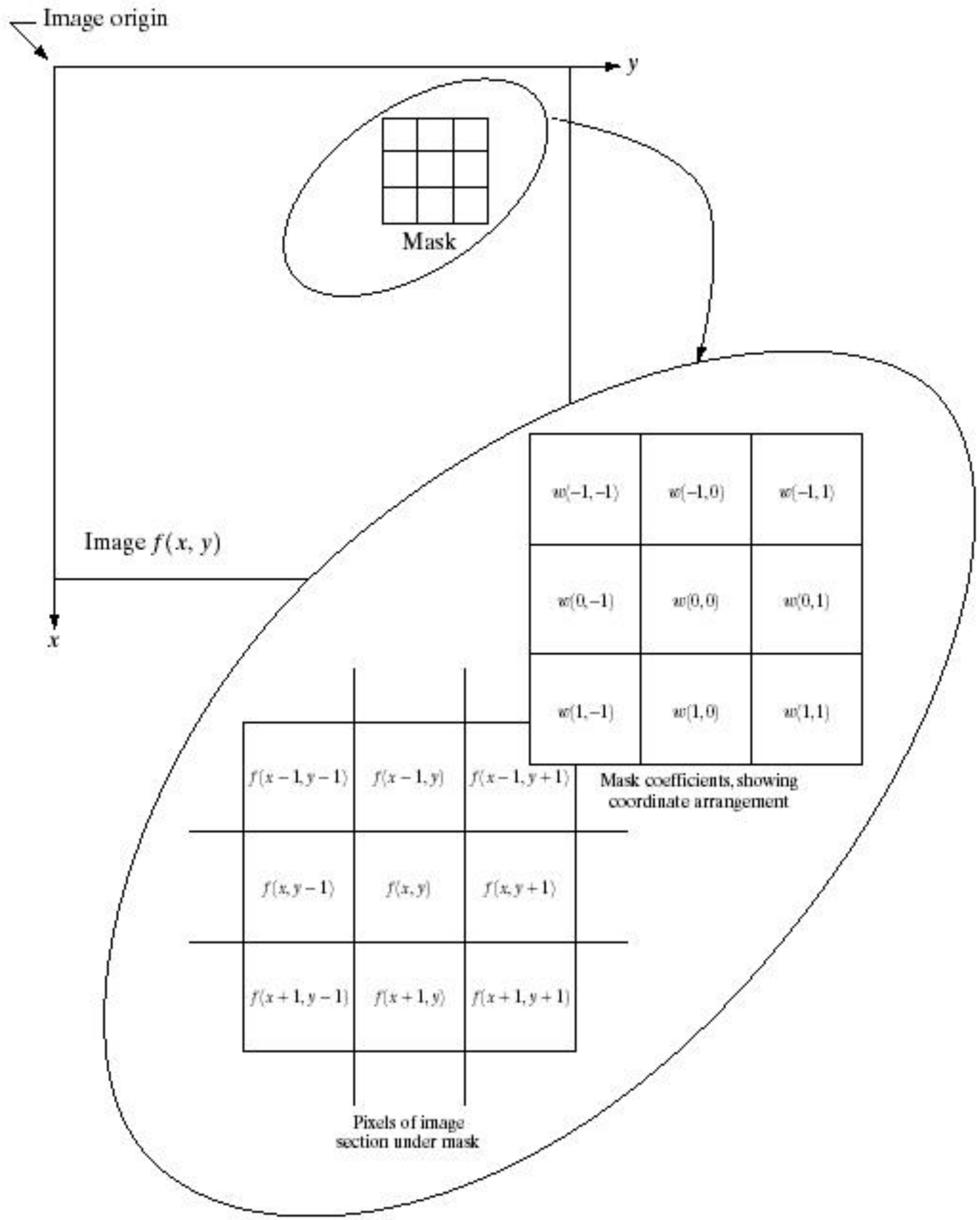
Convolution

$$\sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} f(i,j) w(x-i, y-j)$$

- $w(x-i, y-j)$ est la transposée de (*w* translatée au point (x,y))
- On multiplie "point par point" *f* et la transformée de *w*
- On somme le tout

Convolution

- *w* est généralement petite par rapport à *f*
- *w* est appelée "masque " ou "noyau" (kernel) de convolution"
- les valeurs de *w* sont appelées "coefficients" ou "poids" (*Wheight)*



Convolution

- Question :
  - Qu'obtient-on si on prend pour *w* :

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Convolution

$$\sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} f(i,j)w(x-i,y-j)$$

	x-2	x-1	x	x+1	x+2
y-2	24	32	128	240	255
y-1	12	42	111	154	222
y	4	23	123	176	243
y+1	15	63	145	134	172
y+2	27	12	98	75	143

	-1	0	1
-1	1/9	1/9	1/9
0	1/9	1/9	1/9
1	1/9	1/9	1/9

- Pour $i<x-1$ et $i>x+1$ (idem pour *j*), *w* est nulle
- On retrouve le filtre moyenneur

Convolution

Problème des bords

$$\sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} f(i,j)w(x-i,y-j)$$

- Théoriquement on somme à l'infini
- En pratique les images sont finies
- Pour le masque on considère que les valeurs sont nulles en dehors de son domaine de définition

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

$\Rightarrow$

0	0	0	0	0
0	1/9	1/9	1/9	0
0	1/9	1/9	1/9	0
0	1/9	1/9	1/9	0
0	0	0	0	0

Convolution
Problème des bords

- Pour l'image, il y a plusieurs solutions :

Compléter par des 0

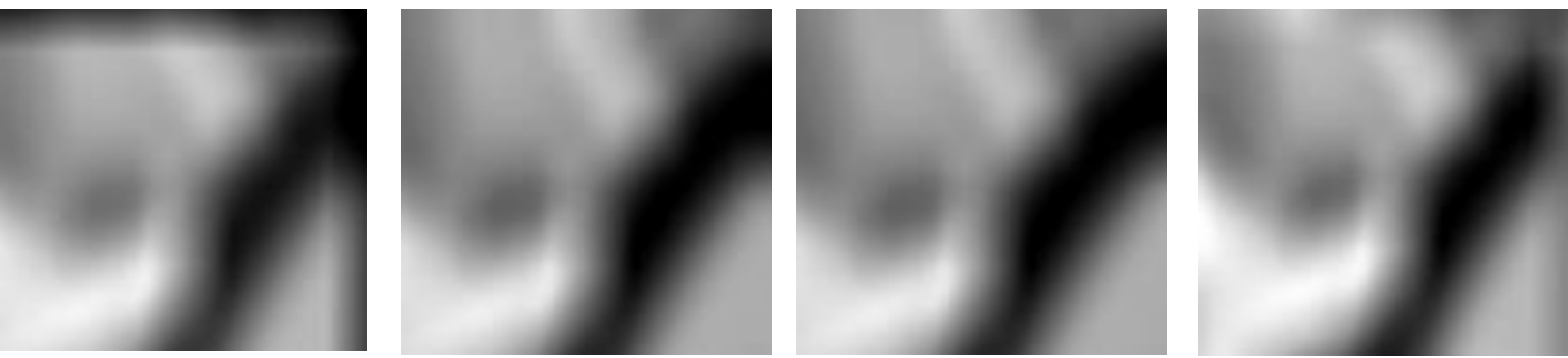
Miroir

Réplication des bords

Domaine cyclique

Convolution
Problème des bords

Coin supérieur droit après application d'un filtre moyenneur



Convolution
Exercice

- On dispose d'une image f de largeur L pixels et de hauteur H pixels.
- Donnez la définition de l'image f' qui correspond à l'extension de f sur le plan $\mathbb{Z}^2$en utilisant:
  - La complétion par 0
  - La réplication des bords
  - La répétition cyclique
  - Le répétition par miroir
- Ex pour la complétion par 0, on a:

$$\forall x, y \in \mathbb{Z}^2, f'(x,y) = \begin{cases} 0 \; si \; x < 0 \; ou \; y < 0 \; ou \; x \geq L \; ou \; y \geq H \\ f(x,y) \; sinon \end{cases}$$

Convolution

Propriétés algébriques

- Commutatif

$$f * w = w * f$$

- Associatif

$$f * (g * w) = (f * g) * w$$

- Distributif

$$f * (g + w) = (f * g) + (f * w)$$

- Associatif par rapport à la multiplication

$$\forall a \in \mathbb{R},\ \ (af) * g = f * (ag) = a(f * g)$$

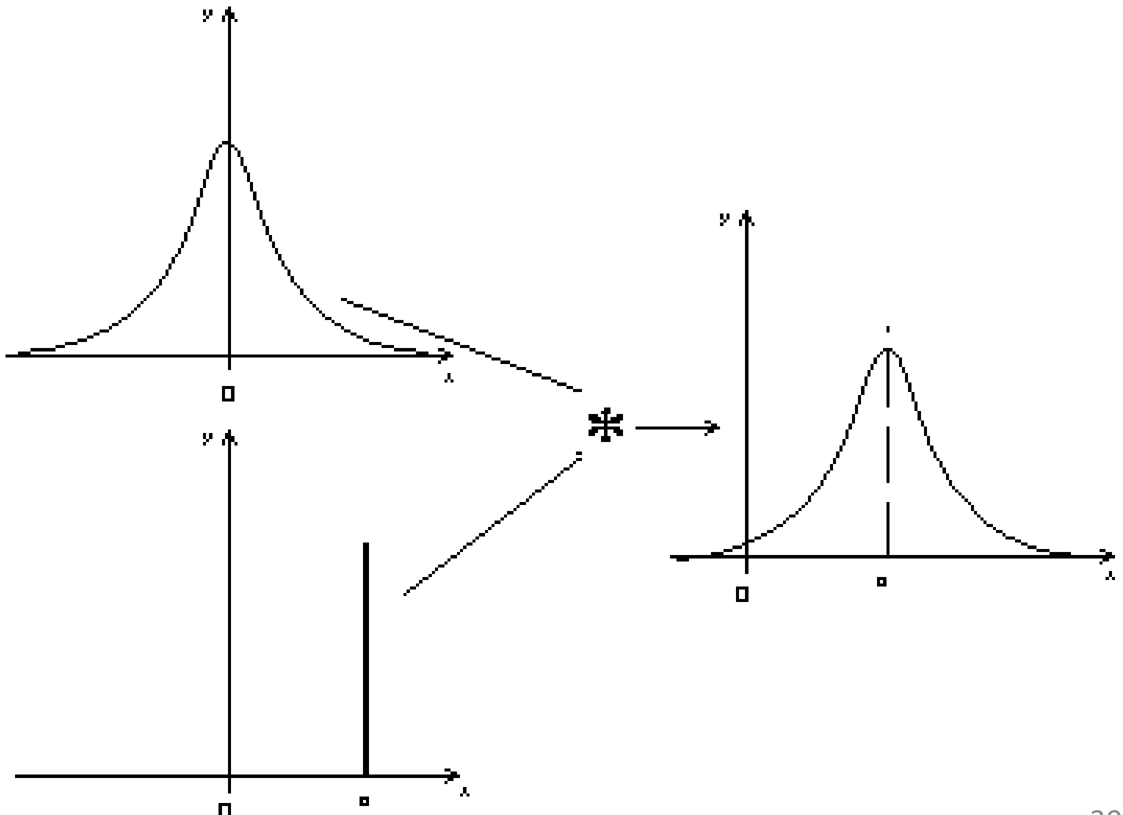
Convolution
Exercice

- Montrer que la convolution est :
  - une application linéaire
  - commutative
  - distributive
  - associative par rapport à la multiplication

  - associative

Convolution et filtre à réponse impulsionnelle

- En signal, un filtre linéaire et invariant temporellement correspond à un filtre de convolution
  - Sa réponse impulsionnelle correspond au masque de convolution



Convolution et transformée de Fourier

- La convolution est intimement liée à la transformée de Fourier
- Si on note $\hat{f}$ la transformée de Fourier de *f*

$$\hat{f}(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(u,v)\exp^{-2i\pi(ux+vy)}\,dudv$$

- On a la relation suivante :

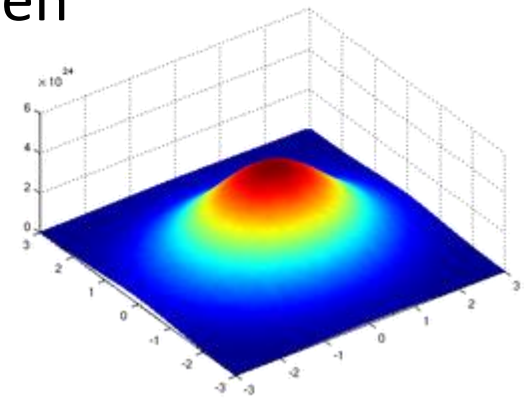
$$\widehat{f * w} = \hat{f}.\hat{w}$$

- Plus de détails dans le cours "Signal et Image"

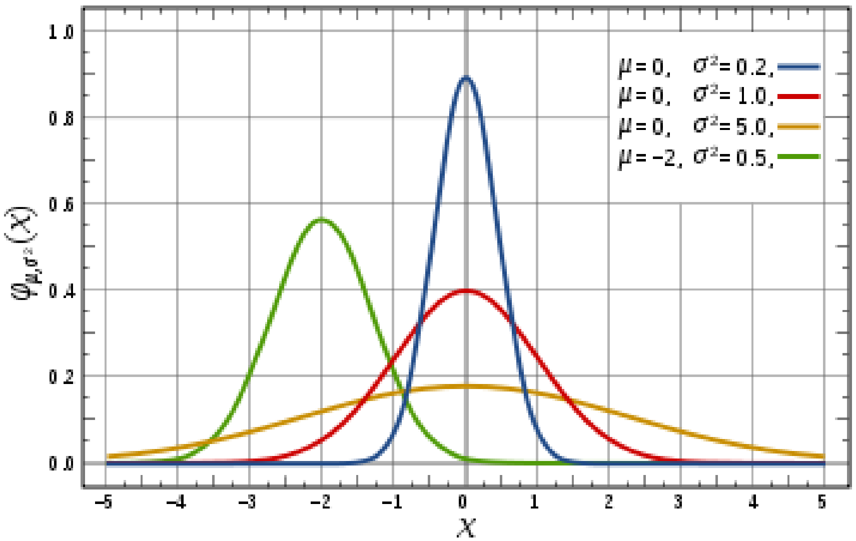
Smoothing

- Smoothing = lissage
- Le filtre moyenneur est un filtre lissant
- En général on utilise plutôt un noyau gaussien

$$w(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

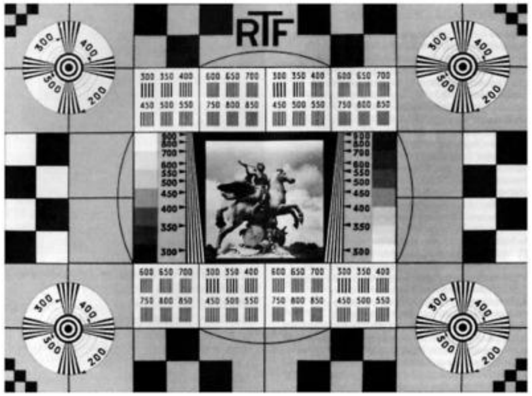


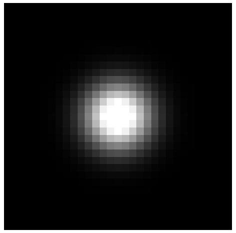
- Le paramètre σ contrôle la "force" du filtre



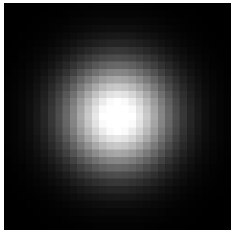
Smoothing

Filtre gaussien pour
différentes valeurs de σ

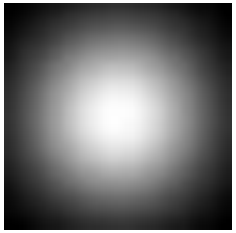




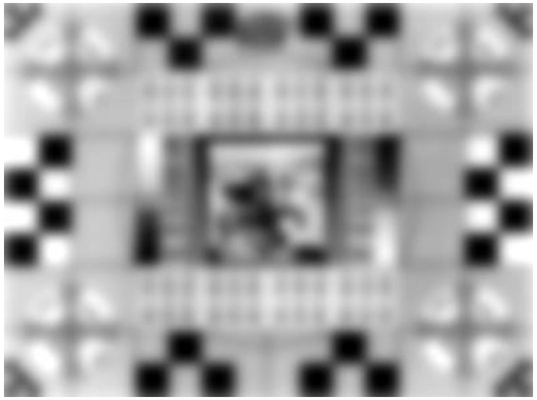
σ = 3

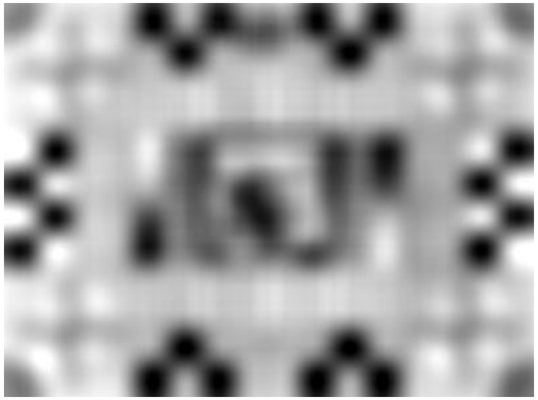


σ = 5



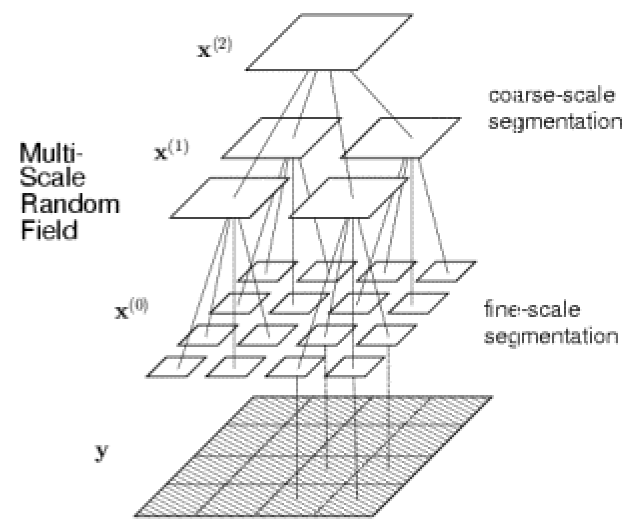
σ = 9





Smoothing

- Une étape de smoothing est importante lorsque qu'on diminue la résolution d'une image (éviter les moirées)
- Ex : construction d'une pyramide d'images
  1. Lissage avec noyau gaussien
  2. Diviser la résolution par deux
  3. Si le résultat fait plus d'un pixel recommencer en 1

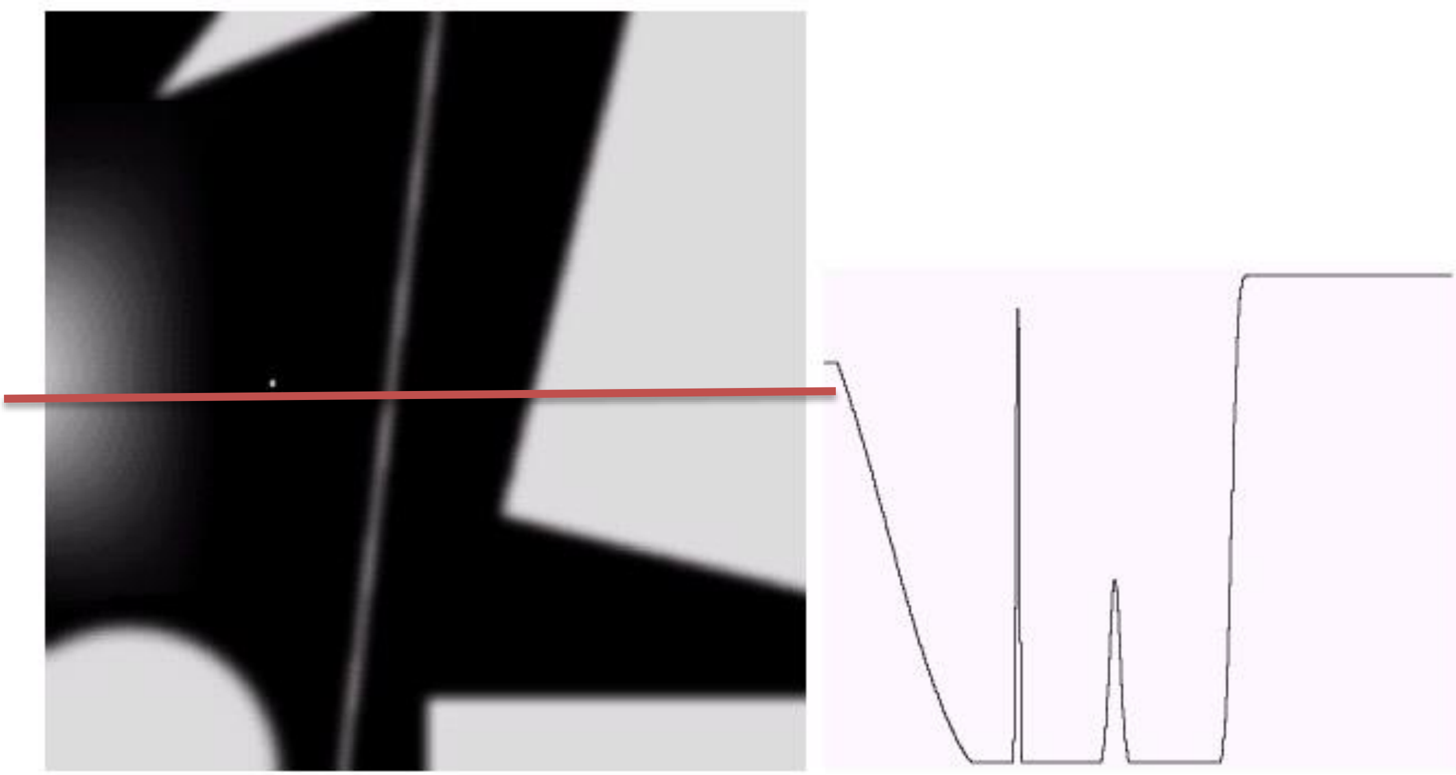


Sharpening

- But : accroitre les détails
- C'est l'opposé du lissage
- Le lissage correspond à une intégration (somme)
- Le sharpening est obtenu avec le traitement inverse : la dérivation

Sharpening

- Pour simplifier : fonction 1D



Sharpening

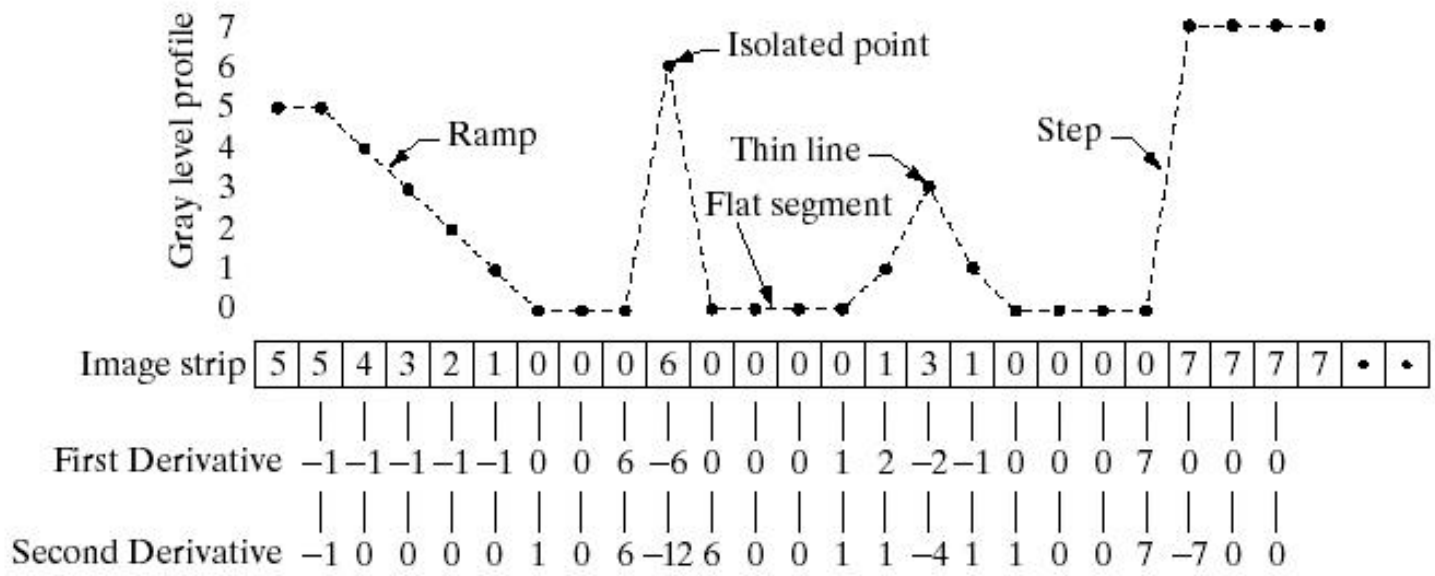
- Comme les images ne sont pas continues, on utilise des approximation de la dérivée
- Soit une fonction *f(x)*
- On définit la dérivée première par

$$\frac{\partial f}{\partial x} := f(x+1) - f(x)$$

- On définit la dérivée seconde par

$$\frac{\partial^2 f}{\partial x^2} := f(x+1) + f(x-1) - 2f(x)$$

Sharpening



- En général :
  - les contours donnés par f' sont plus épais
  - la réponse de f'' aux détails fins est plus forte
  - la réponse de f' au palier est plus forte
  - f'' a une réponse double au palier

Sharpening
Laplacien

- Application en image
- En général on utilise la dérivée seconde
- Dans le cas d'une image 2D, on considère le Laplacien :

$$\Delta^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

Sharpening

Laplacien

$$\Delta^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

- On peut développer

$$\frac{\partial^2 f}{\partial x^2} := f(x+1,y) + f(x-1,y) - 2f(x,y)$$

$$\frac{\partial^2 f}{\partial y^2} := f(x,y+1) + f(x,y-1) - 2f(x,y)$$

- Donc :

$$\Delta^2 f = f(x+1,y) + f(x-1,y) + \\ f(x,y+1) + f(x,y-1) - 4(x,y)$$

Sharpening

Exercice

- Calculer $\frac{\partial^2 f}{\partial x^2}, \ \frac{\partial^2 f}{\partial y^2} \ \text{et} \ \Delta^2 f$ sur l'exemple ci-dessous (on suppose qu'il y a des 0 en dehors de l'image)

1	1	8	9	8
2	2	8	8	8
3	3	8	8	8
4	4	0	12	0
5	5	0	0	0

Sharpening

Laplacien

$$\Delta^2 f = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4(x,y)$$

- Ce qui correspond à la convolution de l'image avec le masque

0	1	0
1	-4	1
0	1	0

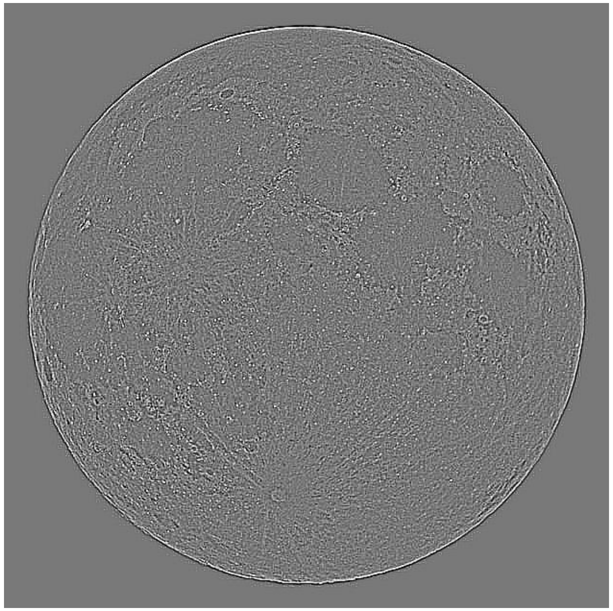
Sharpening

Laplacien

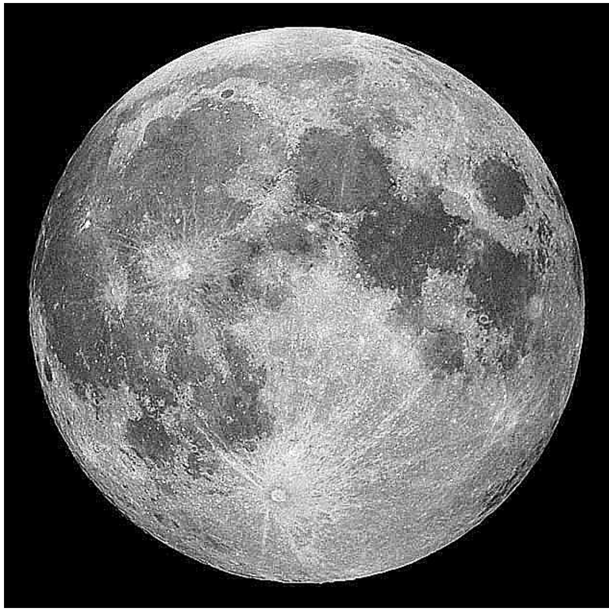
- Le filtre est finalement obtenu en ajoutant le laplacien à l'image originale

$$g = f - \Delta^2 f$$

f



Laplacien



g

Sharpening
Laplacien

- Remarque : au final on a

$$
\begin{aligned}
g(x,y) &= f(x,y) - \Delta^2 f \\
&= f(x,y) - \frac{\partial^2 f}{\partial x^2} - \frac{\partial^2 f}{\partial y^2} \\
&= f(x,y) - (f(x+1,y) + f(x-1,y) - 2f(x,y)) \\
&\qquad - (f(x,y+1) + f(x,y-1) - 2f(x,y))) \\
&= 5f(x,y) - f(x+1,y) - f(x-1,y) \\
&\qquad - f(x,y+1) - f(x,y-1)
\end{aligned}
$$

Sharpening
Laplacien

- Ce qui revient à la convolution avec le masque

0	-1	0
-1	5	-1
0	-1	0

Implémentation

- Temps de calcul
- On considère une image de n*n pixels
- On prend un voisinage de m*m pixels

- Pour chaque pixel : il faut
  - m*m additions
  - m*m multiplications

- Complexité pour l'approche naïve : O(n*n*m*m)

Implémentation
optimisation

- Cas des noyaux séparables
  - Le noyau peut s'écrire comme le produit du vecteur colonne *a* et du vecteur ligne *b*

$$\begin{bmatrix} w_{1,1} & \cdots & w_{1,m} \\ \vdots & & \vdots \\ w_{n,1} & \cdots & w_{n,m} \end{bmatrix} = \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} \begin{bmatrix} b_1 & \cdots & b_m \end{bmatrix}$$

- Le produit d'un vecteur colonne et d'un vecteur ligne et équivalent à une convolution

Implémentation
optimisation

- Comme la convolution est associative on peut réécrire

$$f * w = f * (a * b) = (f * a) * b$$

- On a transformé une convolution 2D en 2 convolutions 1D (comme *a* et *b* sont des vecteurs)

Implémentation
optimisation

- Exemple, prenons

$$w = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

	x-2	**x-1**	**x**	**x+1**	**x+2**
y-2	24	32	128	240	255
y-1	12	42	111	154	222
y	4	23	123	176	243
y+1	15	63	145	134	172
y+2	27	12	98	75	143

La 1ere convolution donne :

- g(x-1,y)=42+2*23+63=151
- g(x,y)=111+2*123+145=502
- g(x+1,y)=154+2*176+134=640

Implémentation

optimisation

$$w = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

	x-2	**x-1**	**x**	**x+1**	**x+2**
y-2					
y-1					
y		151	502	640	
y+1					
y+2					

La 2ème convolution donne :

- h(x,y)=151+2*502+640=1795

On trouve bien le même résultat qu'avec une convolution avec *w*

- *h(x,y)=42+2*23+63+111*2+4*123+ 2*145+154+2*176+134=*1795

Implémentation
optimisation - exercice

- Complexité
- Pour une image de taille n*n
- Pour un masque de taille k*l
- La décomposition donne un vecteur a de taille k et un vecteur b de taille l

- Quelle est la complexité de l'opération de convolution
  – Sans séparation du noyau ?
  – Avec séparation du noyau ?

Implémentation
optimisation

- Lorsque w n'est pas séparable on peut généraliser la méthode en utilisant la décomposition en valeurs singulières

Implémentation
optimisation

- On a vu que la convolution est équivalent à une multiplication dans le domaine fréquentiel (après une transformée de Fourier)
- Les algorithmes de transformée de Fourier rapide (FFT) ont une complexité en O(n*n*log(n)) pour une image de dimension n*n
- Donc les différentes étapes sont:
  - 2 FFT de complexité O(n*n*log(n))
  - 1 multiplication point par point O(n*n)
  - 1 FFT inverse O(n*n*log(n))
- La complexité finale est en O(n*n*log(n))
  - Ne dépend pas de la taille du masque : intéressant si il est grand

Implémentation

Exercice

- Quel gain en nombre de multiplications pour une image 1000*1000 et un masque 9*9 en utilisant la technique du noyau séparable ?
- Et en utilisant la FFT?
- Mêmes questions si le masque est de taille 100*100

Filtre

1. Introduction
   a) Exemple : filtre moyenneur
   b) Filtre global, semi-local, local et adaptatif

2. Filtre linéaire - Convolution
   a) Formulation
   b) Smoothing
   c) Sharpening
   d) Implémentation

3. Filtre non linéaire
   a) Filtre de rang, filtre médian
   b) Morphologie binaire
   c) Morphologie en niveau de gris

4. Binarisation

Filtre non linéaire

Filtre de rang

- On garde l'idée d'une fenêtre glissante sur l'image
- On ne s'intéresse plus à une combinaison linéaire des valeurs des pixels dans la fenêtre
- On va plutôt chercher le k-ième plus grand (ou plus petit)
- On parle de *ranking*

Filtre non linéaire

Filtre de rang - médian

- En général on s'intéresse au médian
  - dans une liste triée de *m* nombre, l'élément médian est celui qui se trouve au milieu
- Ex on considère la liste (1 3 6 9 223)
  - L'élément médian est 6
- Un des intérêts du median par rapport à la moyenne (ou tout autre combinaison linéaire) : sa robustesse
  - dans l'ex précédent la moyenne est 48,4, influencée majoritairement par l'élément 223 qui semble être un outlier

Filtre non linéaire

Médian - Exercice

- Montrez que le filtre médian n’est pas un filtre linéaire.

Filtre non linéaire

Filtre de rang - médian

- Détail de calcul
  - Pour une fenêtre de 3*3 autour du pixel

24	32	128	240	255
12	42	111	154	222
4	23	123	176	243
15	63	145	134	172
27	12	98	75	143

→

		123		

$$23 \leq 42 \leq 63 \leq 111 \leq 123 \leq 134 \leq 145 \leq 154 \leq 176$$

Filtre non linéaire

Filtre de rang - médian

- Les filtres de rangs ne créent pas de nouvelle valeur dans l'image
- Le filtre médian est particulièrement efficace pour réduire le bruit sel-et-poivre

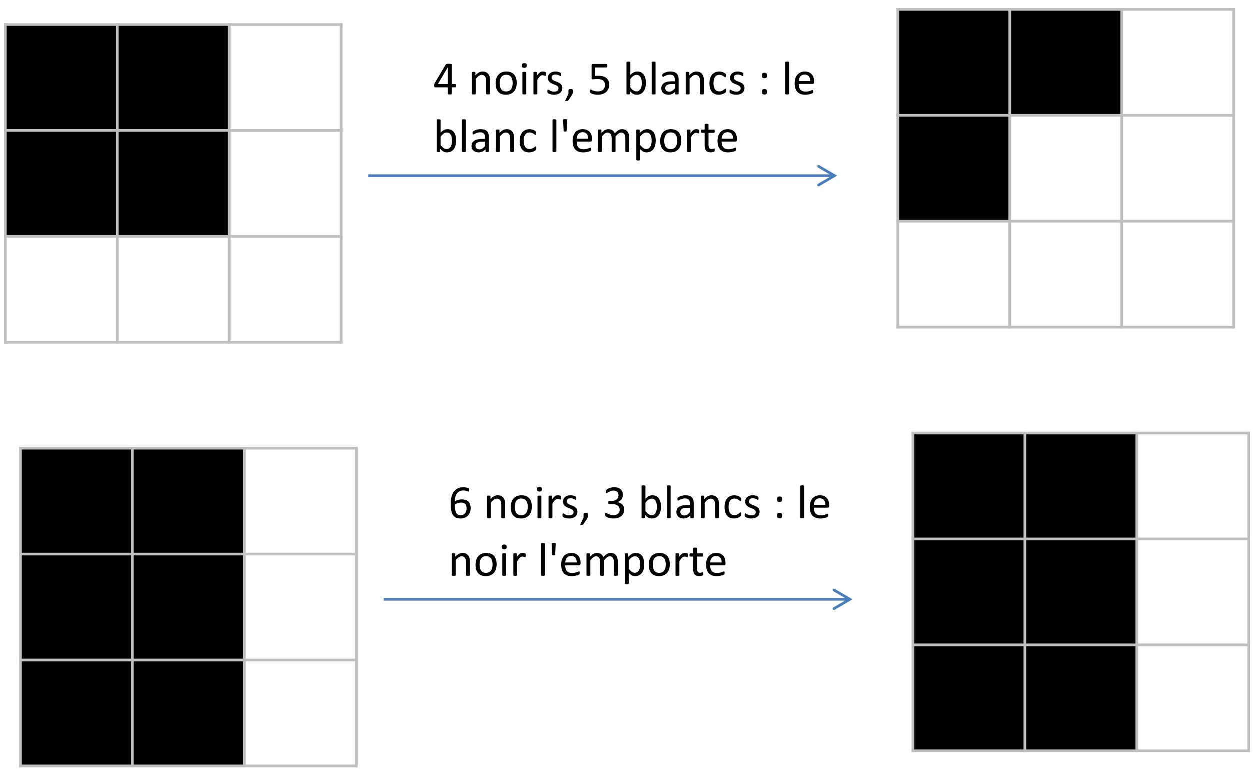
Originale

Bruitée sel et poivre (10%)

Filtre médian 3*3

Filtre non linéaire
Filtre de rang

- Dans le cas des images binaires, le filtre médian revient à faire un vote à la majorité



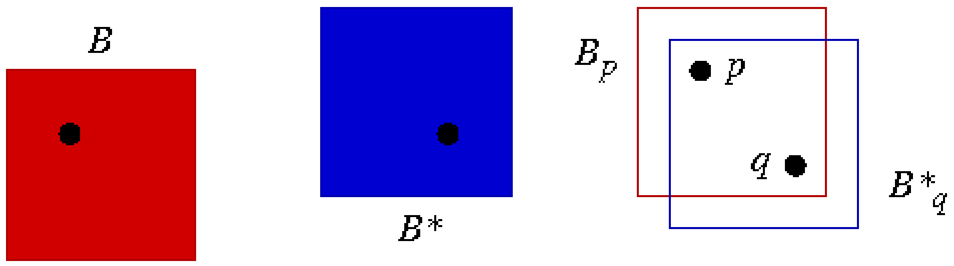
Morphologie
Binaire

- Idée générale de la morphologie binaire
  - les filtres linéaires reposent sur l'addition et la multiplication
  - La morphologie binaire repose sur l'union et l'intersection ensembliste

- On s'intéresse ici à la morphologie dite "structurelle"
  - Les opérateurs dépendent d'un élément structurant
  - L'élément structurant décrit la forme primitive d'intérêt : carré, cercle, segment, ...

Morphologie
Binaire

- Rappel sur les opérations ensemblistes
  - Union : $A \cup B$
  - Intersection : $A \cap B$
  - Complément : $A^c = \{p : p \notin A\}$
  - Différence : $A/B = A \cap B^c$
  - Réflexion : $\breve{B} = B^* = \{p : p = -b \quad \forall b \in B\}$
  - Translation : $A_z = \{p : p = a + z \quad \forall a \in A\}$

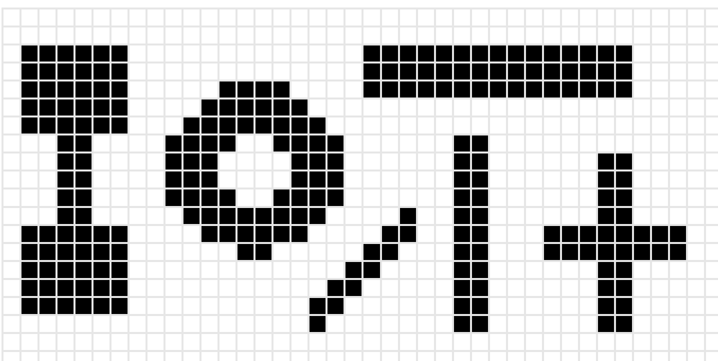


Morphologie

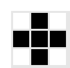
Binaire - Dilatation

- Dilatation de A (image) par B (élément structurant):

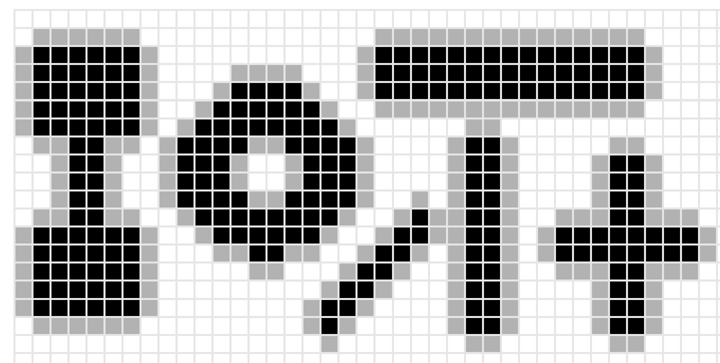
$$\delta_B(A) = A \oplus B = \{x \ : \ (\breve{B})_x \cap A \neq \emptyset\} = \bigcup_{b \in B} A_b$$



$\oplus$



=



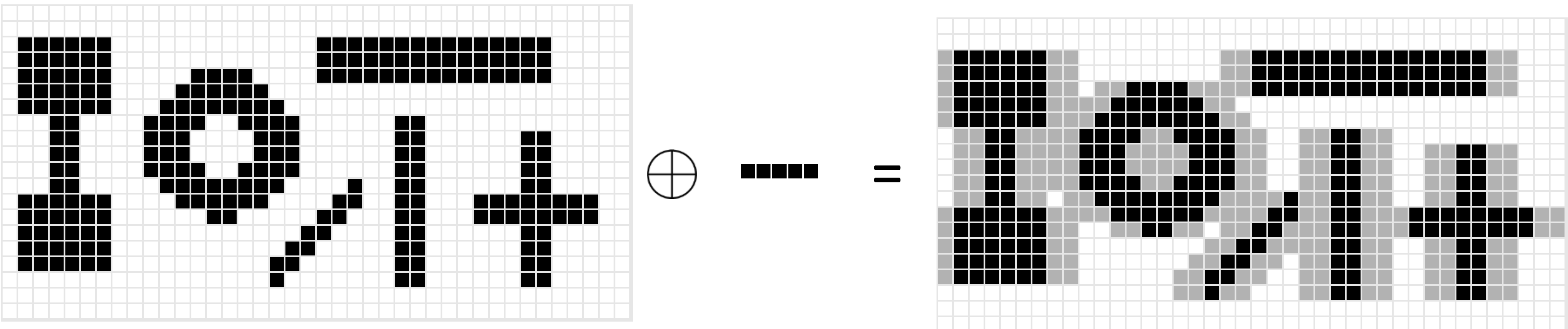
image

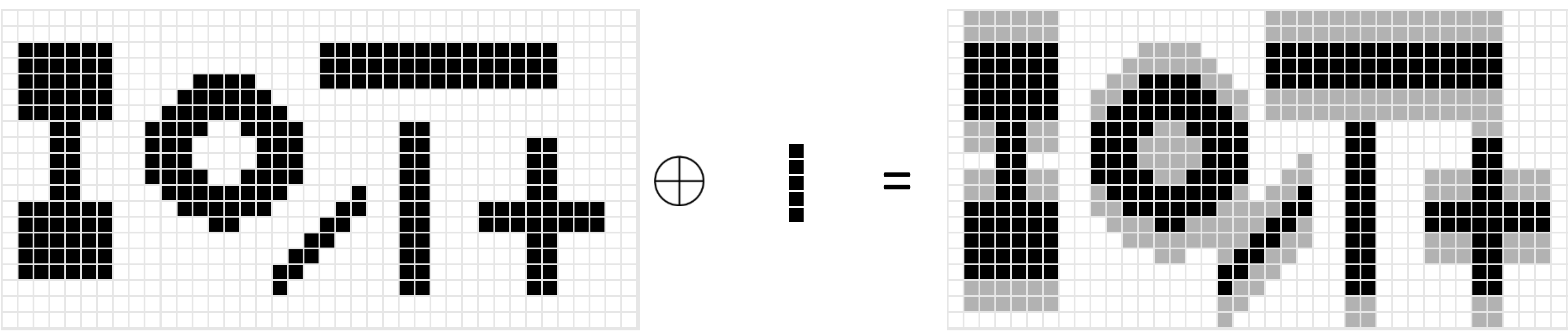
élément structurant

dilatation (en gris et noir)

Morphologie

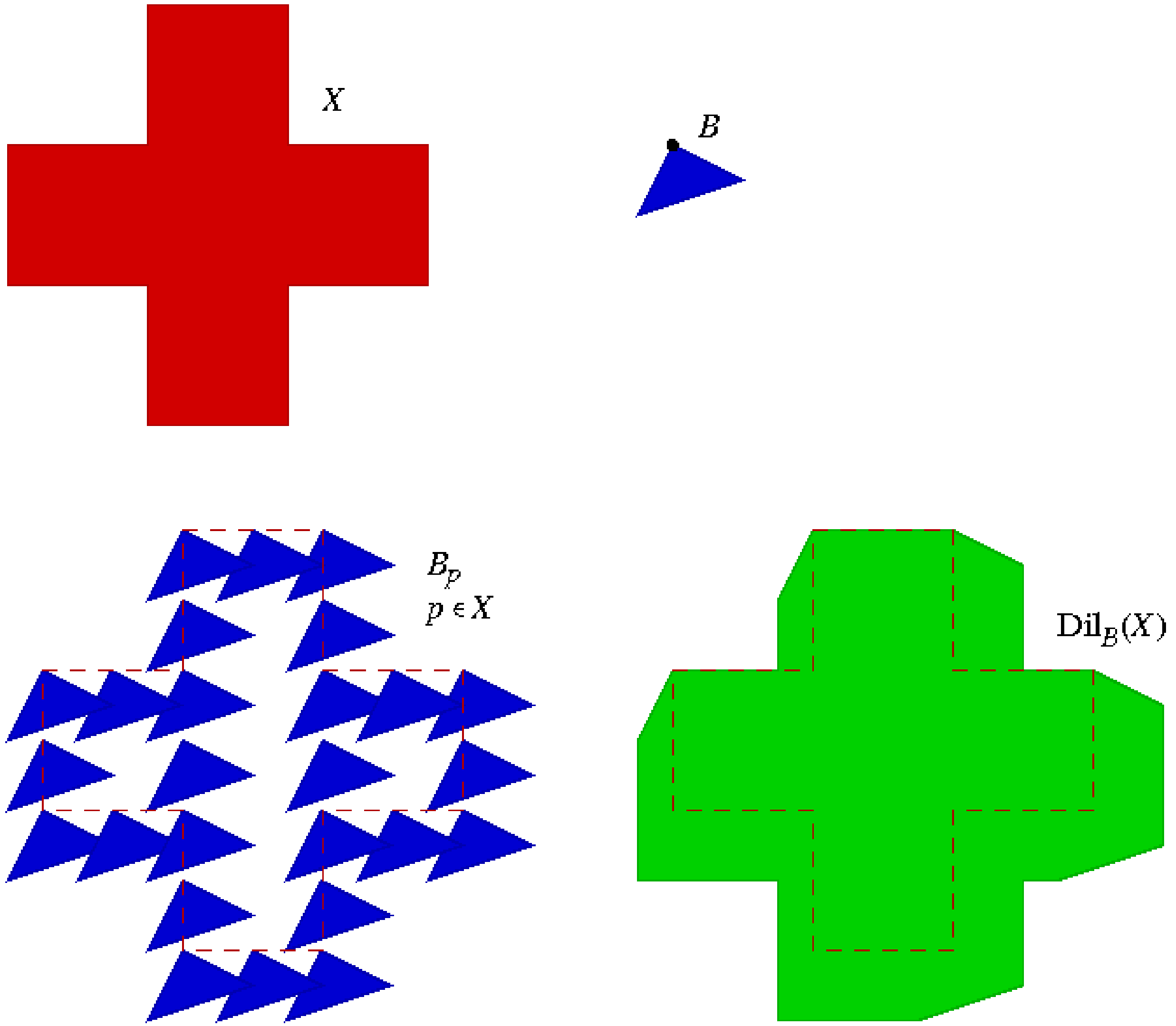
Binaire - Dilatation





Morphologie

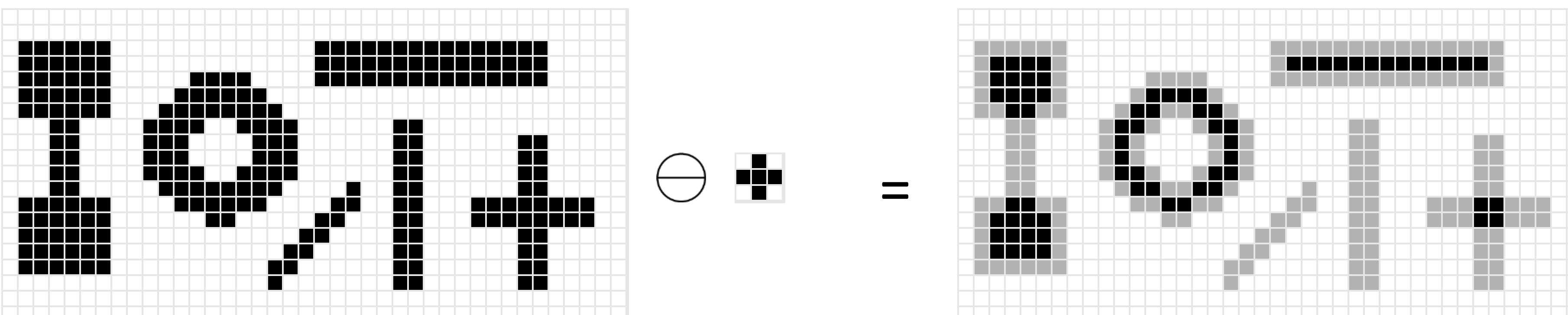
Binaire - Dilatation



Morphologie
Binaire - Erosion

- Erosion de A (image) par B (élément structurant):

$$\varepsilon_B(A) = A \ominus B = \{x \;:\; B_x \subseteq A\} = \bigcap_{b \in B} (A)_{-b}$$



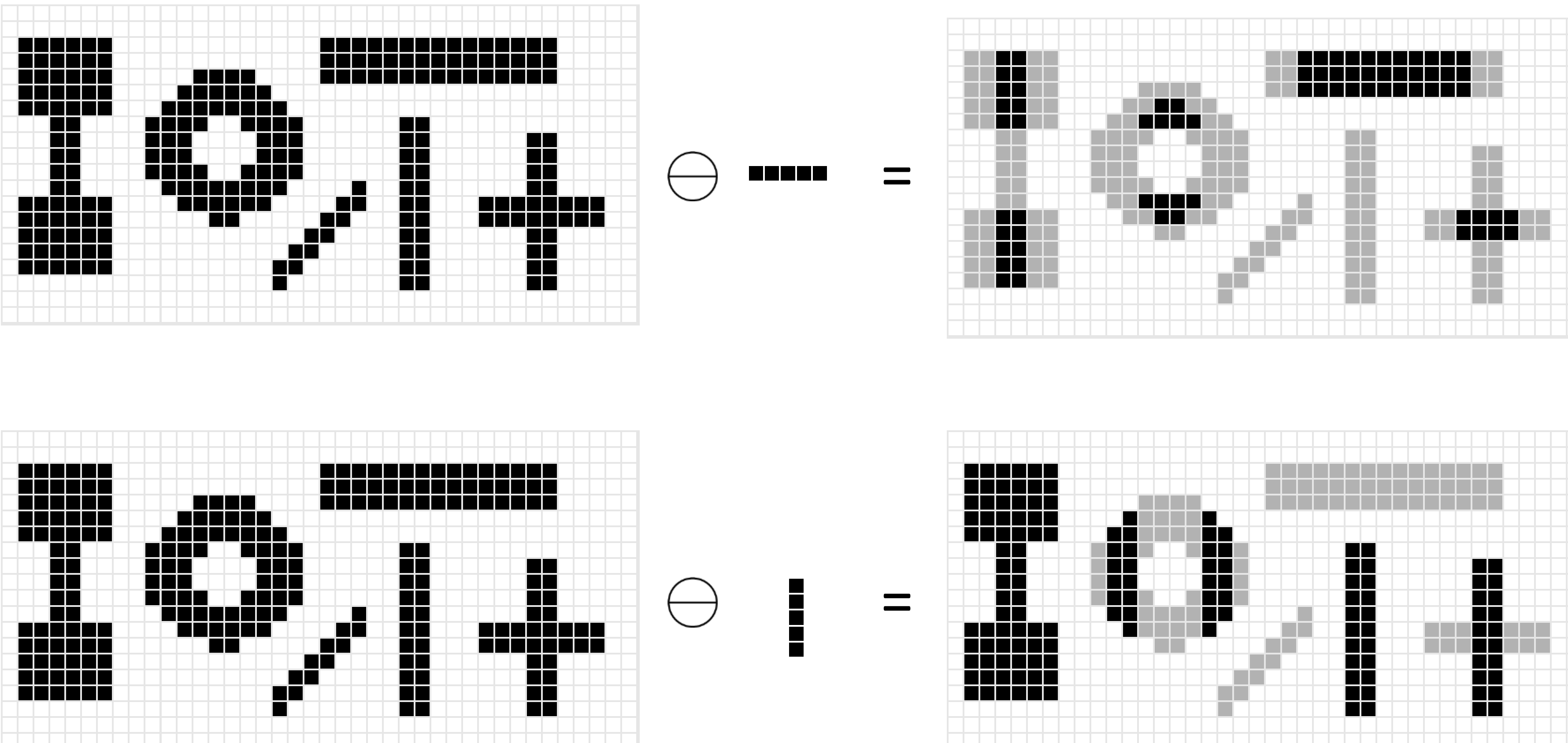
image

élément structurant

érosion (en noir)

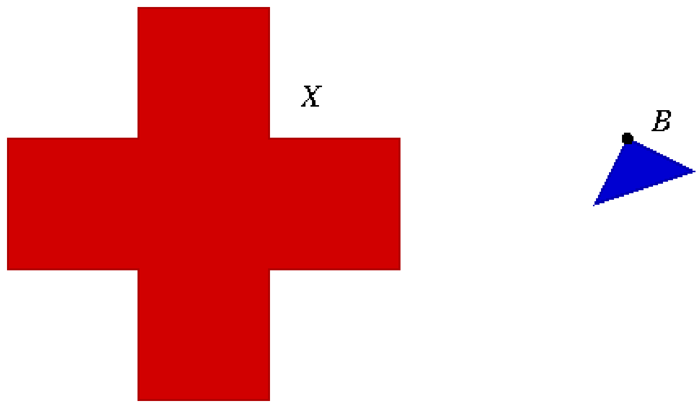
Morphologie

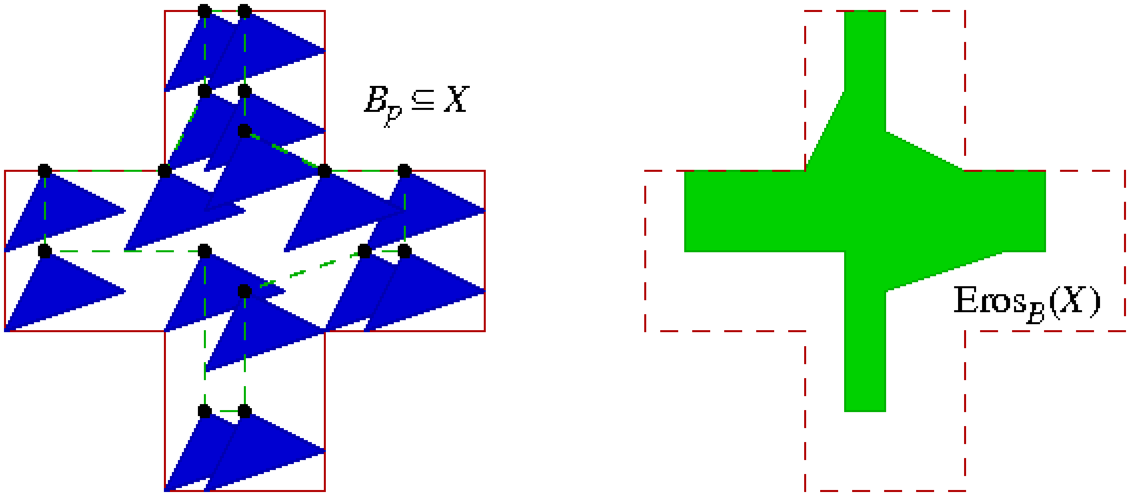
Binaire - Erosion



Morphologie

Binaire - Erosion





Morphologie
Binaire – Erosion et Dilatation

- Propriétés :
  - dilatation et érosion sont des opérateurs croissants :

$$A \subseteq B \Rightarrow \delta_C(A) \subseteq \delta_C(B)$$

$$A \subseteq B \Rightarrow \epsilon_C(A) \subseteq \epsilon_C(B)$$

  - Si l'élément l'origine est dans l'élément structurant : $(0,0) \in B$
    - la dilatation est extensive $A \subseteq \delta_B(A)$
    - l'érosion est anti-extensive $\epsilon_B(A) \subseteq A$
  - La dilatation commute avec l'union

$$\delta_C(A \cup B) = \delta_C(A) \cup \delta_C(B)$$

  - L'érosion commute avec l'intersection

$$\epsilon_C(A \cap B) = \epsilon_C(A) \cap \epsilon_C(B)$$

Morphologie
Binaire – Erosion et Dilatation

- Exercice: montrez que:
  - La dilatation est une opération croissante
  - Si (0,0) appartient à l'élément structurant alors la dilation est une opération extensive et donnez un exemple où la dilatation n'est pas une opération extensive
  - Montrez que la dilatation commute avec l'union

Morphologie
Binaire – Erosion et Dilatation

- Propriétés :
  - la dilatation est commutative : $A \oplus B = B \oplus A$
  - la dilatation est associative : $A \oplus (B \oplus C) = (A \oplus B) \oplus C$
  - dilatation et érosion sont duales par complémentation

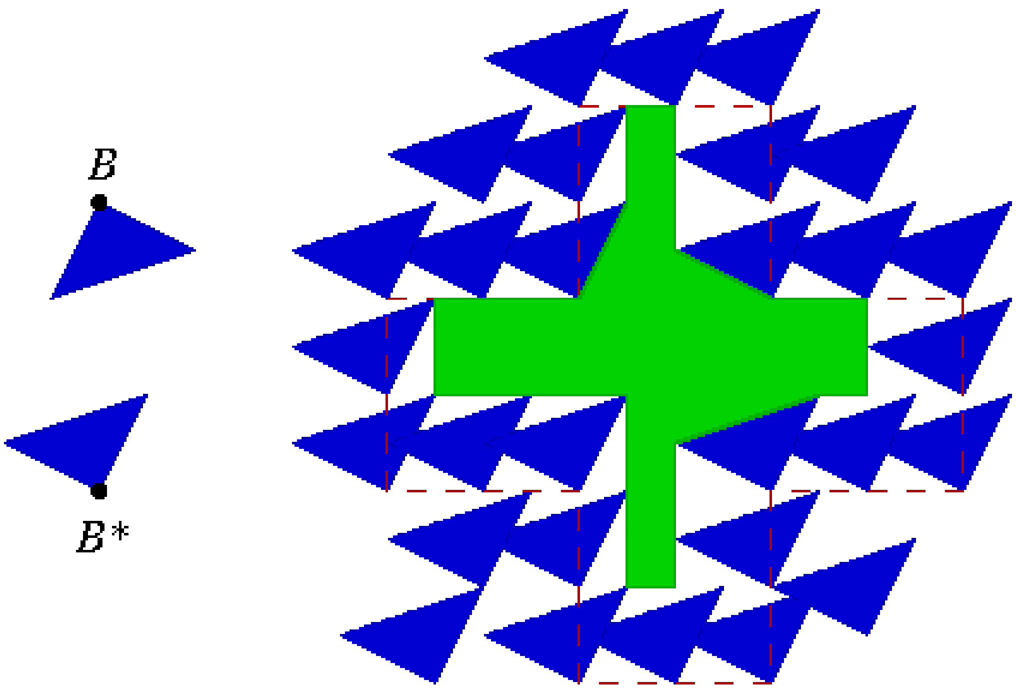
$$\delta_{\breve{B}}(A) = A \oplus \breve{B} = (A^c \ominus B)^c = (\epsilon_B(A^c))^c$$

  - Décomposition de l'élément structurant

$$A \ominus (B \oplus C) = (A \ominus B) \ominus C$$

Morphologie
Binaire – Erosion et Dilatation

Illustration de la dualité entre érosion et dilatation

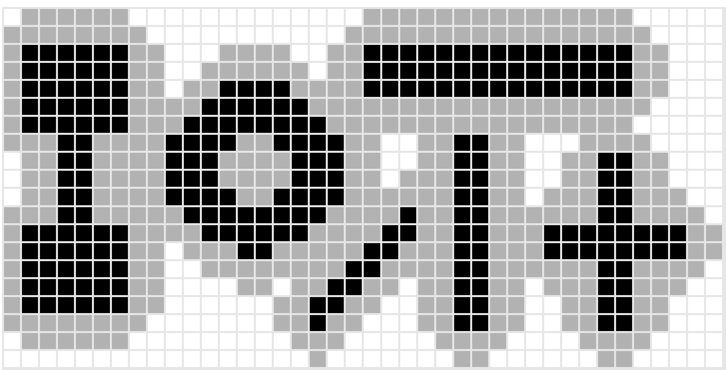


Morphologie

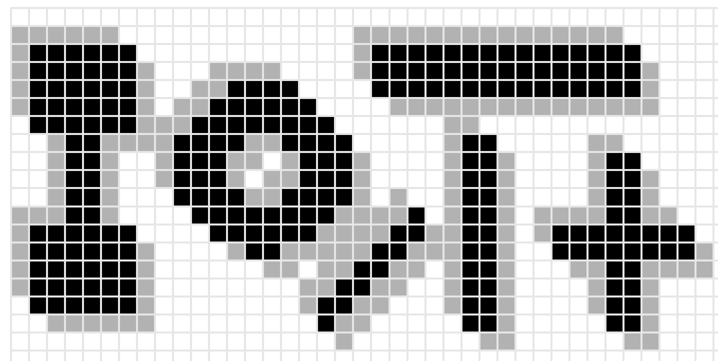
Binaire – Erosion et Dilatation

- Exercice : trouver l'élément structurant utilisé pour chacune des dilatations suivantes :

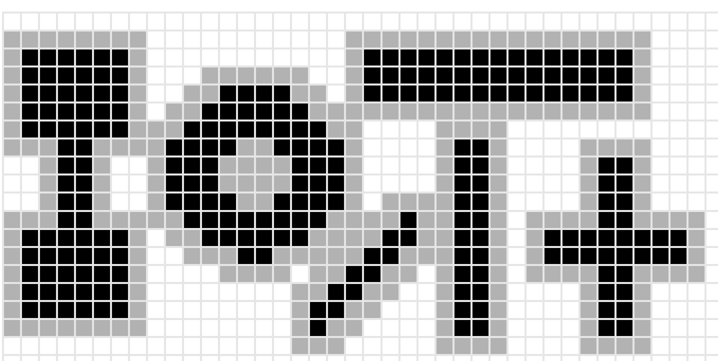
1



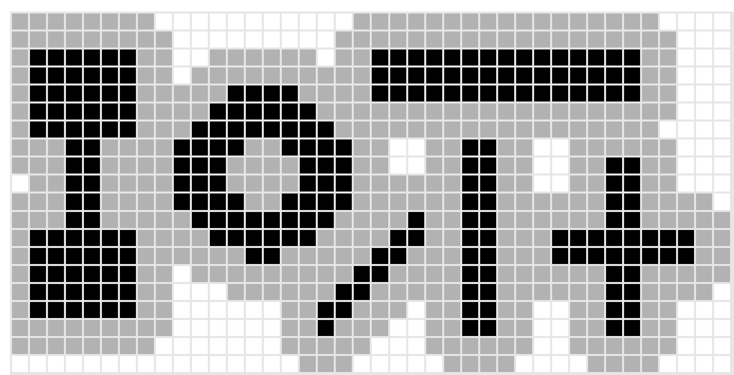
3



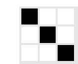
2



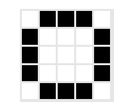
4



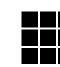
a



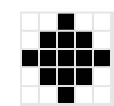
b



c



d



Morphologie

Binaire – Ouverture

- La composition d'une érosion et d'une dilatation donne une ouverture

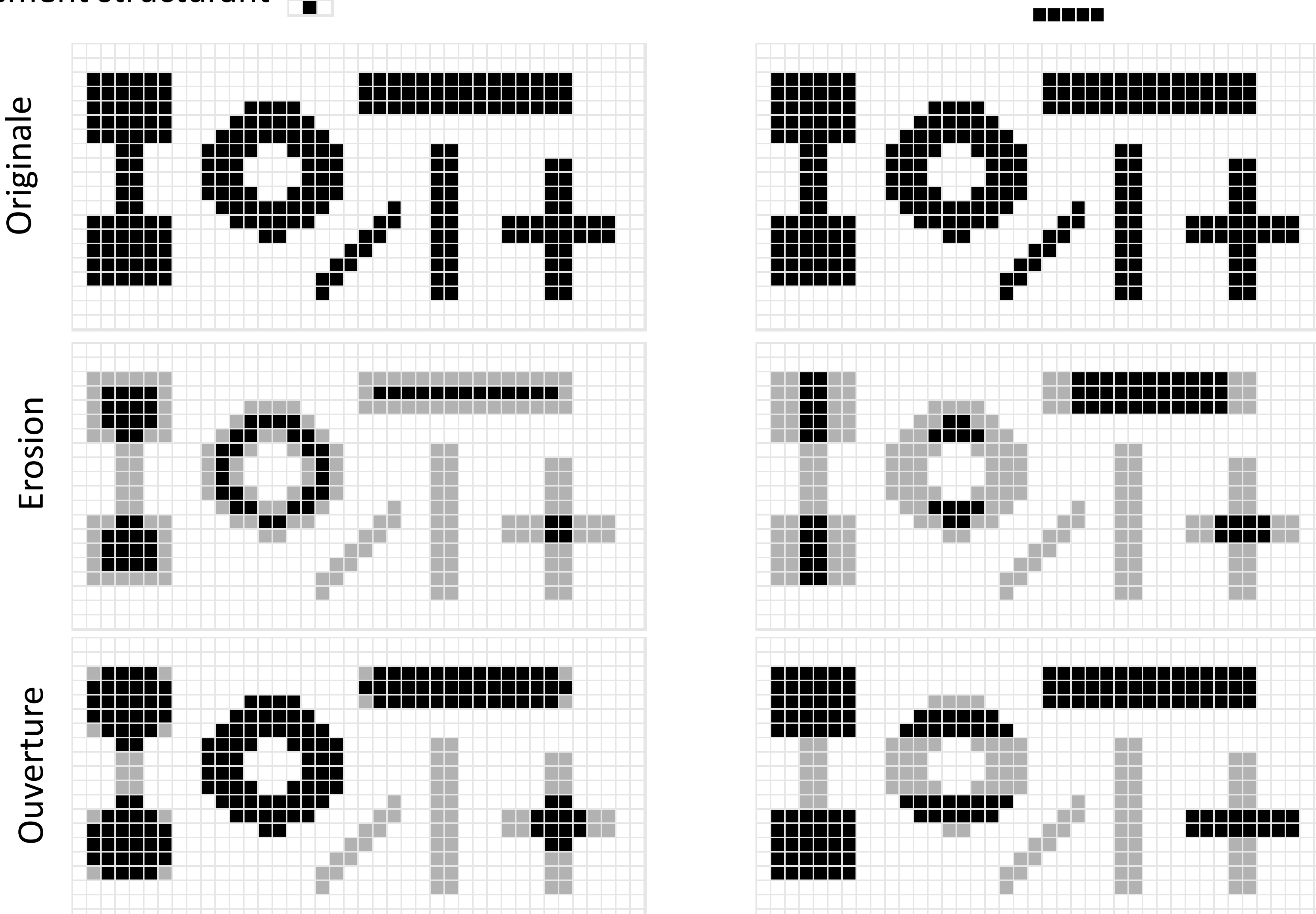
$$\gamma_B(X) = X \circ B = \delta_B(\epsilon_B(X))$$

- L'ouverture supprime les éléments plus petits que l'élément structurant et laisse les autres inchangés

Morphologie

Binaire – Ouverture

Elément structurant



Morphologie

Binaire – Ouverture

- La composition d'une dilatation et d'une érosion donne une fermeture

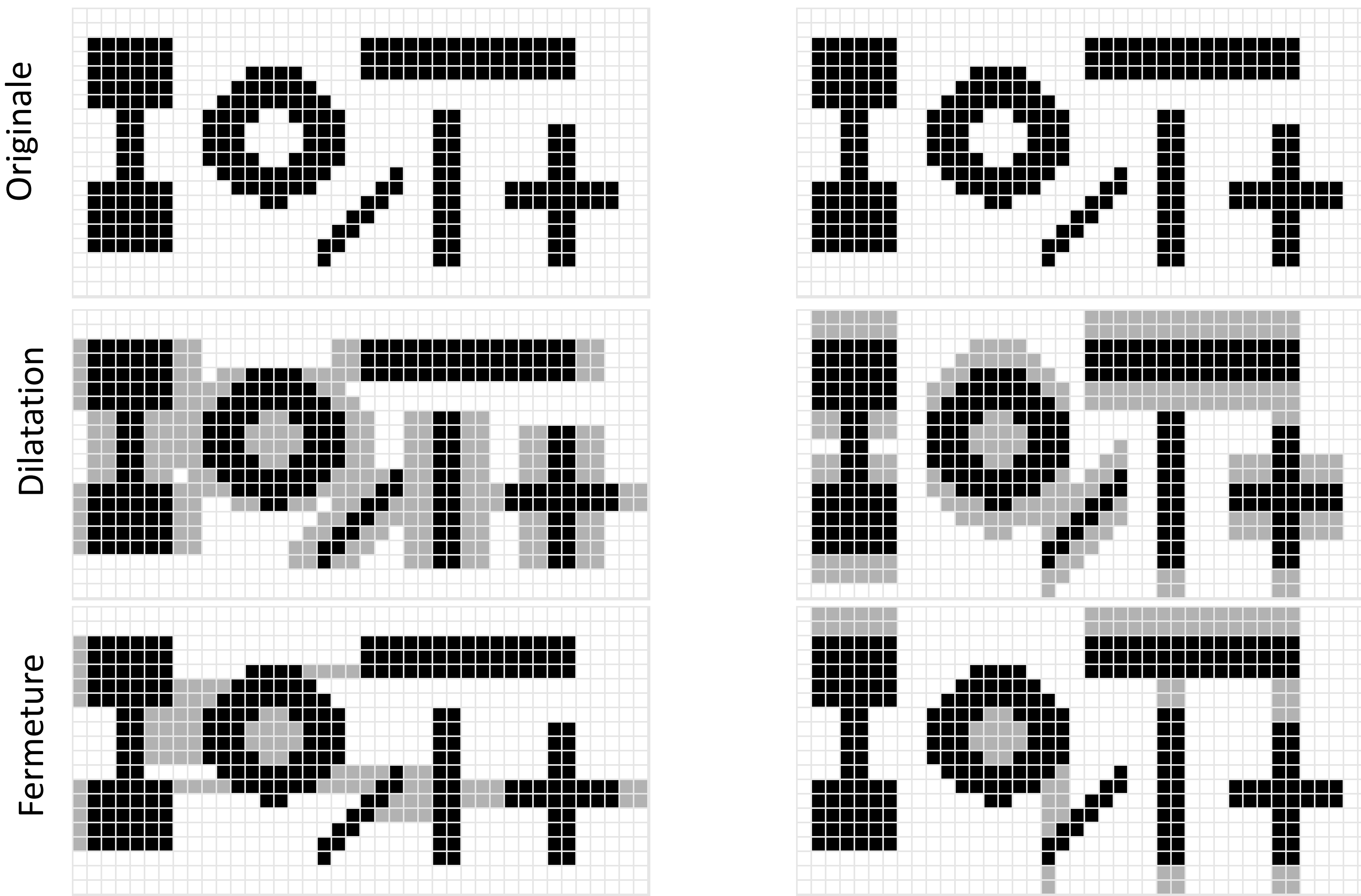
$$\phi_B(X) = X \bullet B = \epsilon_B(\delta_B(X))$$

- La fermeture bouche les trous *plus petit* que l'élément structurant et laisse le reste inchangé

Morphologie

Binaire – Fermeture

Elément structurant



Morphologie

Binaire : Ouverture – Fermeture

- Propriétés de l'ouverture et de la fermeture
  - idempotence :

$$\gamma_B(X) = \gamma_B(\gamma_B(X)), \quad \phi_B(X) = \phi_B(\phi_B(X))$$

  - croissance :

$$X \subseteq Y \Rightarrow \phi_B(X) \subseteq \phi_B(Y)$$
$$X \subseteq Y \Rightarrow \gamma_B(X) \subseteq \gamma_B(Y)$$

  - extensivité de la fermeture

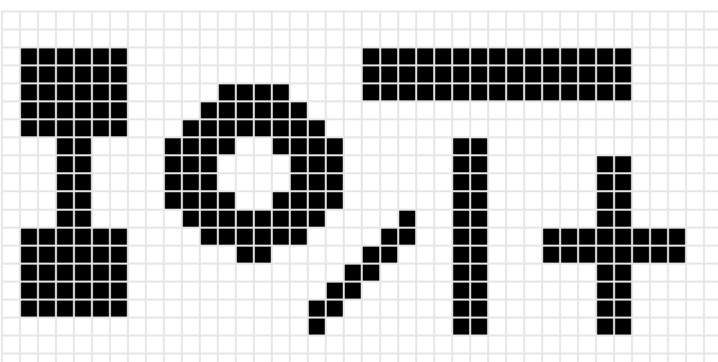
$$X \subseteq \gamma_B(X)$$

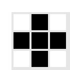
  - anti-extensivité de l'ouverture

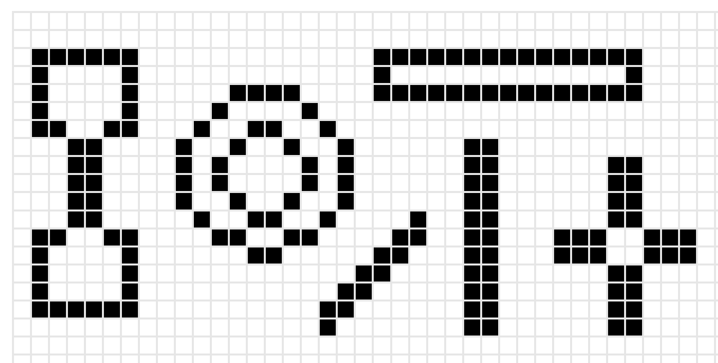
$$\phi_B(X) \subseteq X$$

Morphologie
Binaire - Gradient

- Définition alternative du gradient
  - gradient interne : $A - (A \ominus B)$



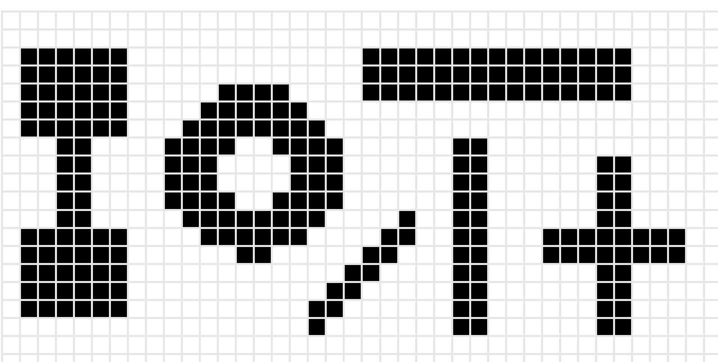


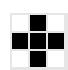


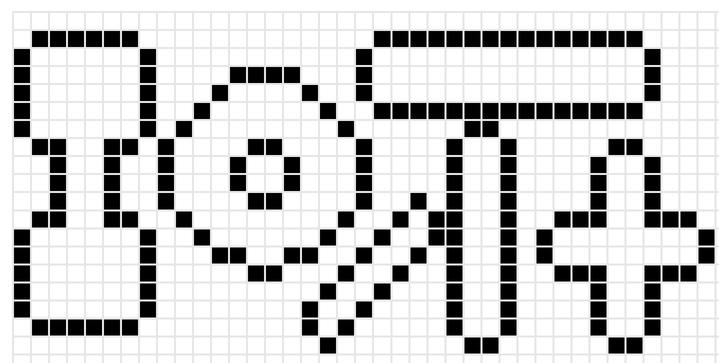
Morphologie

Binaire - Gradient

- Définition alternative du gradient
  - gradient externe : $(A \oplus B) - A$

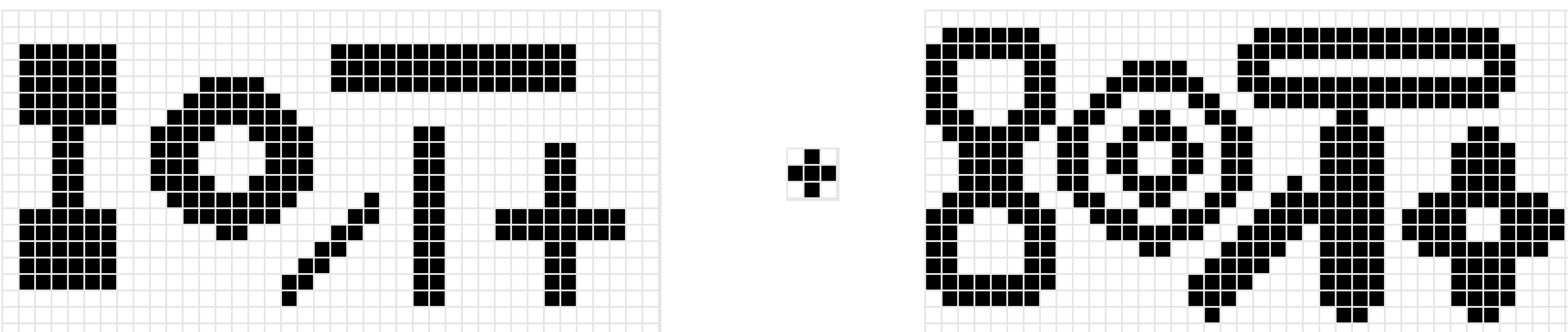






Morphologie
Binaire - Gradient

- Définition alternative du gradient
  - gradient morphologique : $G_B(A) = (A \oplus B) - (A \ominus B)$



Morphologie
Niveau de gris

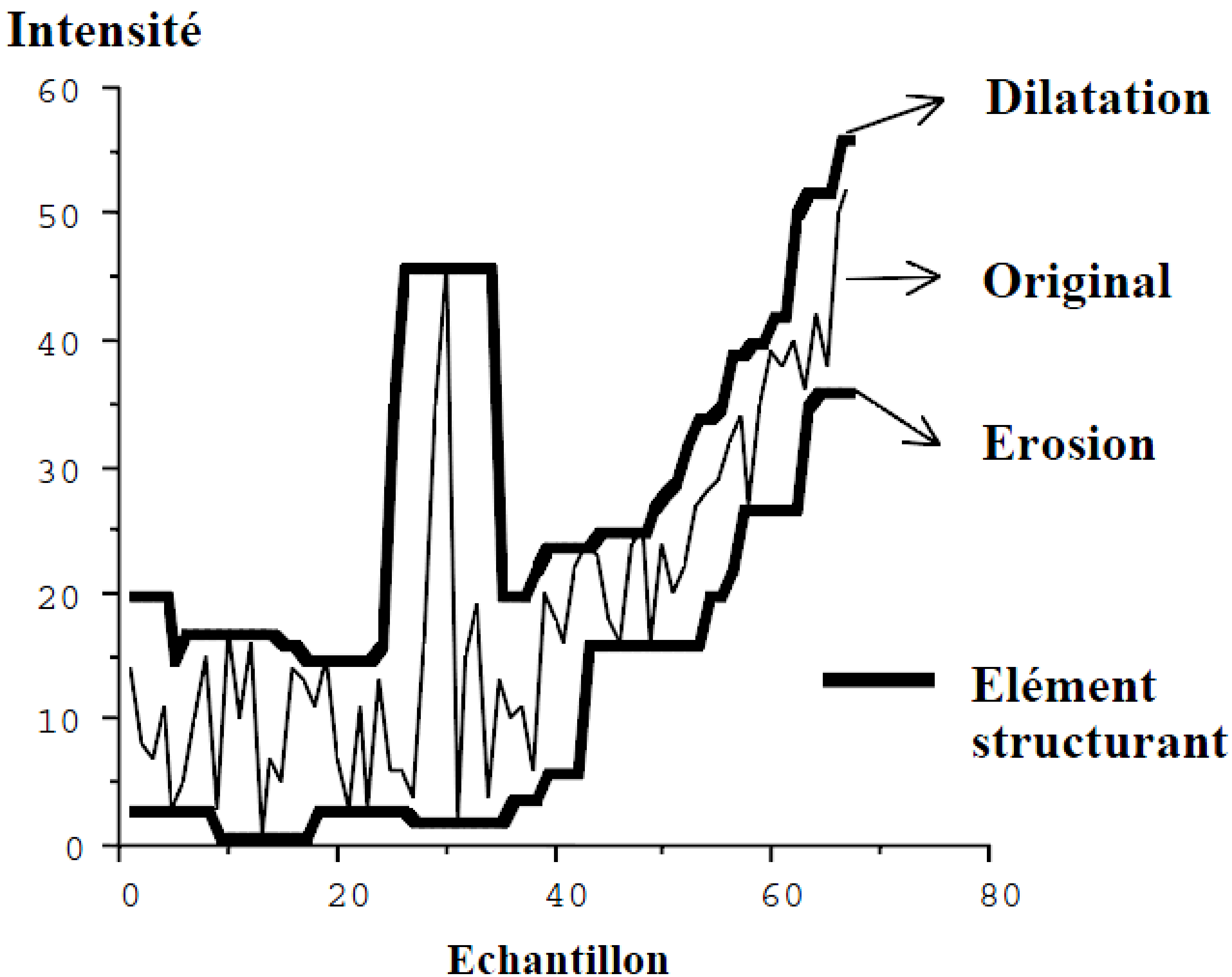
- On considère maintenant des images/fonctions du plan discret dans un sous ensemble fermé de R (un ensemble qui inclut ses bornes sup et inf)
- On définit alors l'érosion et la dilatation par

$$(f \oplus h)(x) = \sup_{y \in h} \left(f(x-y) + h(y) \right)$$

$$(f \ominus h)(x) = \inf_{-y \in h} \left(f(x-y) - h(y) \right)$$

- Tous les opérateurs binaires vus s'étendent naturellement aux images à niveau de gris à partir de l'érosion et de la dilation

Morphologie
Niveau de gris



Morphologie
Niveau de gris

– Exemples : f, $f \ominus D$, $f \oplus D$, $f \circ D$, $f \cdot D$

- Gradient morphologique

Filtre

1. Introduction
   a) Exemple : filtre moyenneur
   b) Filtre global, semi-local, local et adaptatif

2. Filtre linéaire - Convolution
   a) Formulation
   b) Smoothing
   c) Sharpening
   d) Implémentation

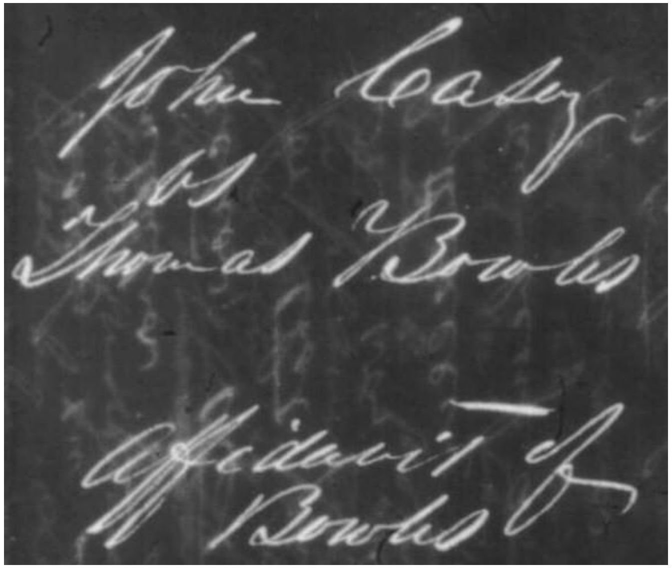
3. Filtre non linéaire
   a) Filtre de rang, filtre médian
   b) Morphologie binaire
   c) Morphologie en niveau de gris

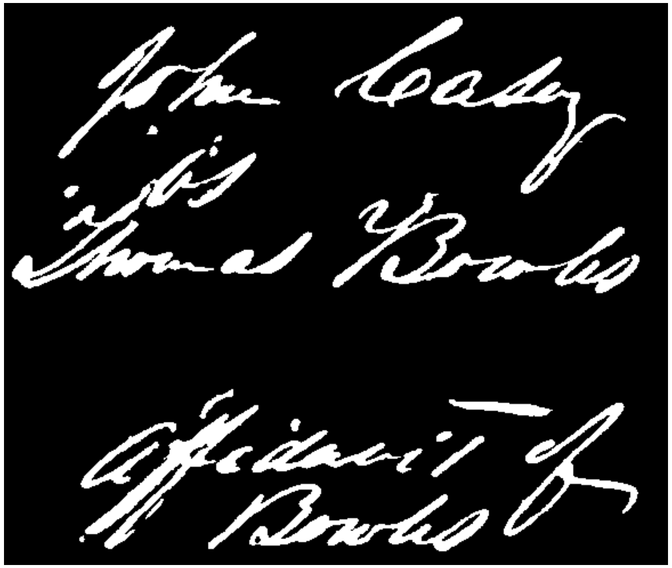
4. Binarisation

Filtre

Binarisation

- Cas particulier, on cherche un filtre qui transforme une image en niveau de gris en une image binaire





Filtre

Binarisation - Seuillage

- Solution la plus simple :
  - le seuillage globale (tresholding)
- On définit un seuil *S*
  - Tous les pixels dont la valeur est inférieur à *S* sont mis à zéro
  - Les autres à 1

$$t(v) = \begin{cases} 0 \text{ si } v < S \\ 1 \text{ sinon} \end{cases}$$

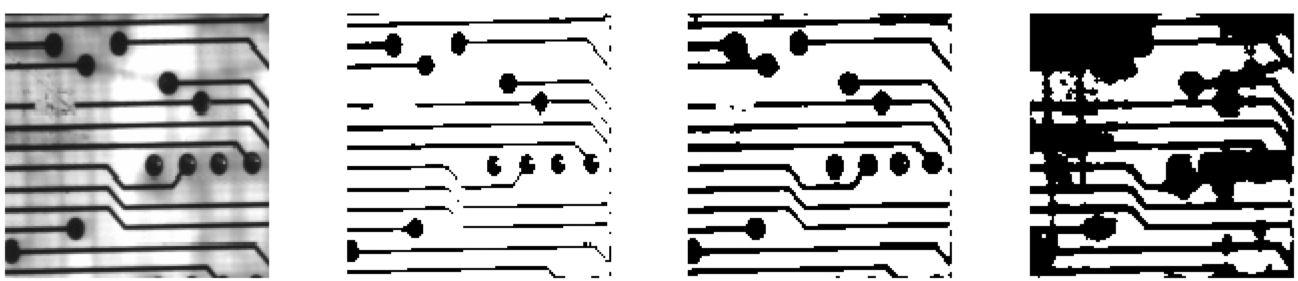
Filtre

Binarisation - Seuillage

- Comment déterminer *S* automatiquement
  - Difficulté variable
  - Dépend du contenu de l'image, des conditions d'illumination
- Formalisation : on cherche à séparer les pixels en deux ensembles (cluster) B (background) et F (foreground) tels que

$$B \cap F = \emptyset \quad \text{et} \quad \forall x \in B,\ \forall y \in F,\ f(x) < f(y)$$

  - c'est un cas particulier de segmentation



Filtre

Binarisation - Seuillage

- L'idée générale est de "regarder" l'histogramme de l'image pour trouver où couper.

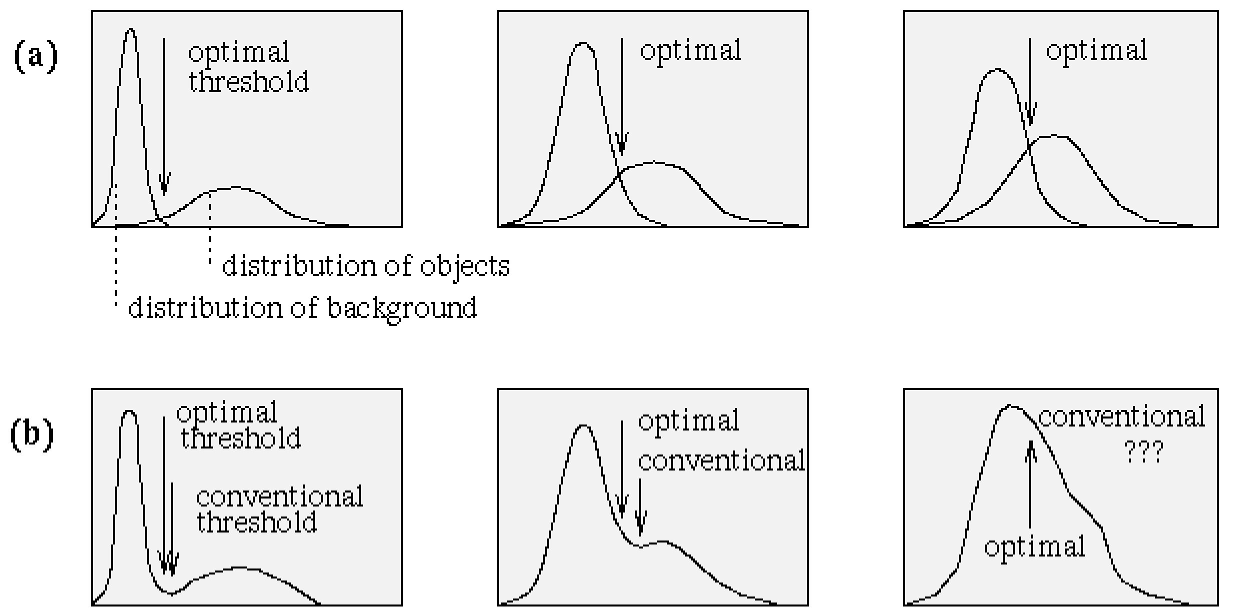


Figure 5.4 *Grey level histograms approximated by two normal distributions; the threshold is set to give minimum probability of segmentation error: (a) Probability distributions of background and objects, (b) corresponding histograms and optimal threshold.*

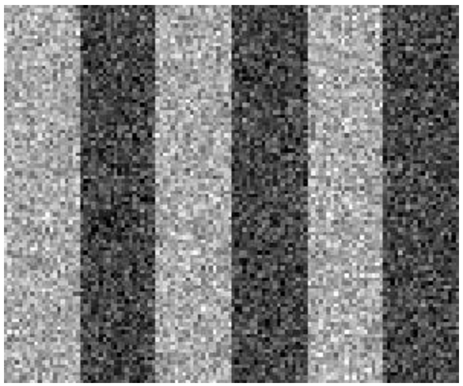
Filtre

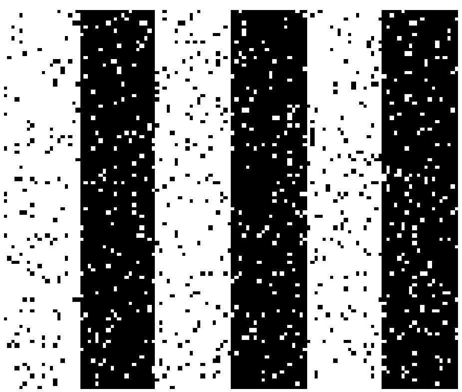
Seuillage – 2-means

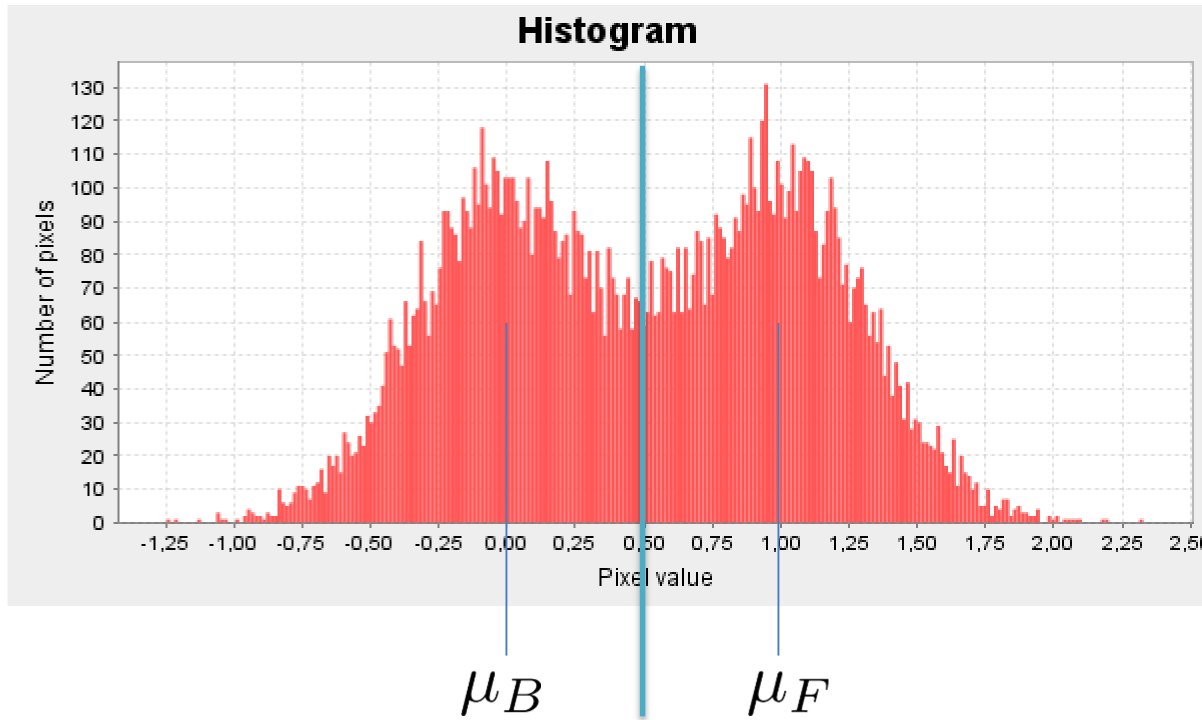
- Idée générale de l'approche 2-means (cas particulier des k-means) :
- On note μ_B (resp μ_F) la valeur moyenne des pixels dans *B* (resp *F*)
- Un point de valeur *v* appartient :
  - à *B* si $|v - \mu_B| < |v - \mu_F|$
  - à *F* si $|v - \mu_B| > |v - \mu_F|$

Filtre

Seuillage – 2-means







$$S = \frac{\mu_B + \mu_F}{2}$$

Filtre

Seuillage – 2-means

- Problème : savoir si un point appartient à un des ensembles demande de connaitre la moyenne des ensembles qui dépendent elles-même de l'appartenance du point à l'un des ensemble...

Filtre
Seuillage – 2-means

- Algorithme itératif d'approximation

Algorithme 1: Seuil2Mean

---

input : Image en niveaux de gris im, valeurs initiales μ_B^0 et μ_F^0
output : Seuil S
$S^0 := (\mu_B^0 + \mu_F^0)/2;$
$i := 0;$
while $S^{i+1} \neq S^i$ **do**
  $\mu_B^{i+1} := \mu_F^{i+1} := 0; \quad nb_B := nb_F := 0;$
  for $Pixel\ p \in im$ **do**
    if $im(p) < S^i$ **then**
      $\mu_B^{i+1} += im(p); \quad nb_B++;$
    else
      $\mu_F^{i+1} += im(p); \quad nb_F++;$
  $\mu_B^{i+1} := \mu_B^{i+1}/nb_B; \quad \mu_F^{i+1} := \mu_F^{i+1}/nb_F;$
  $S^{i+1} := (\mu_B^{i+1} + \mu_F^{i+1})/2;$

[fleqn]

Filtre

Seuillage – Otsu

- La méthode d'Otsu consiste à trouver le seuil S qui minimise la variance intra-classe
- On définit les variances des classes B et F :

$$\sigma_B^2 = \mu_{B^2} - (\mu_B)^2 \quad \text{et} \quad \sigma_F^2 = \mu_{F^2} - (\mu_F)^2$$

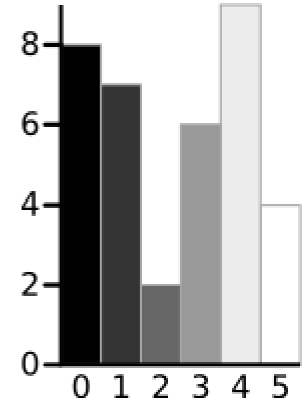
μ_B est la moyenne des valeurs des pixels dans B

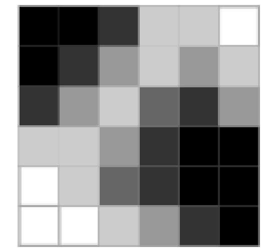
μ_{B^2} est la moyenne des valeurs au carré des pixels dans B

- On définit la variance intra-classe de B et F :

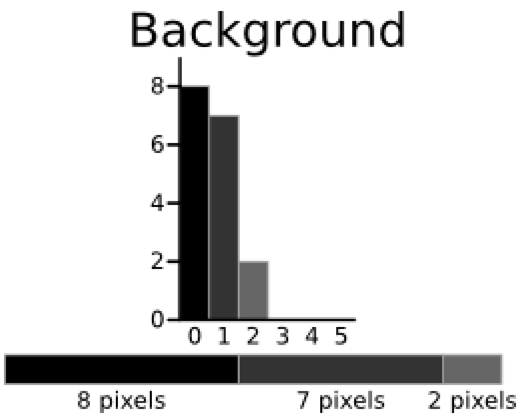
$$\sigma_i^2 = \frac{\mathrm{card}(B) \times \sigma_B^2 + \mathrm{card}(F) \times \sigma_F^2}{\mathrm{card}(B) + \mathrm{card}(F)}$$

Filtre
Seuillage – Otsu





- Ex sur une image à 6 niveaux de gris:
- Pour le seuil S=3



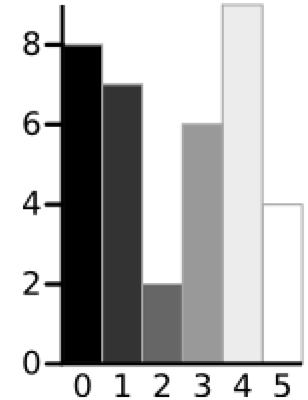
$$\mathrm{card}(B) = 8 + 7 + 2 = 17$$

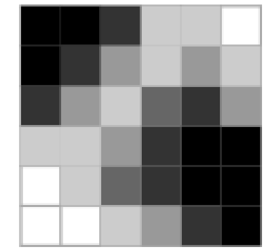
$$\mu_B = \frac{(0 \times 8) + (1 \times 7) + (2 \times 2)}{\mathrm{card}(B)} = 0.6471$$

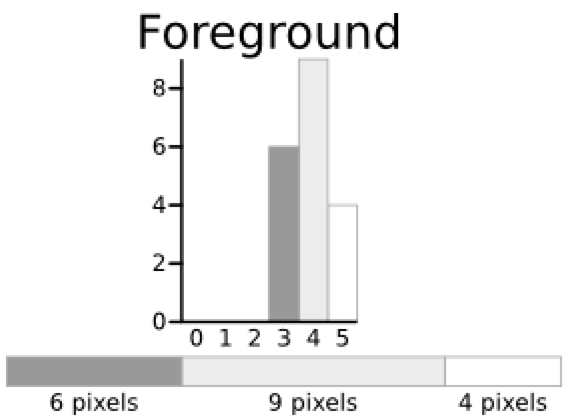
$$\mu_{B^2} = \frac{(0^2 \times 8) + (1^2 \times 7) + (2^2 \times 2)}{\mathrm{card}(B)} = 0.8824$$

$$\sigma_B^2 = \mu_{B^2} - (\mu_B)^2 = 0.4637$$

Filtre
Seuillage – Otsu







$$\mathrm{card}(F) = 6 + 9 + 4 = 19$$

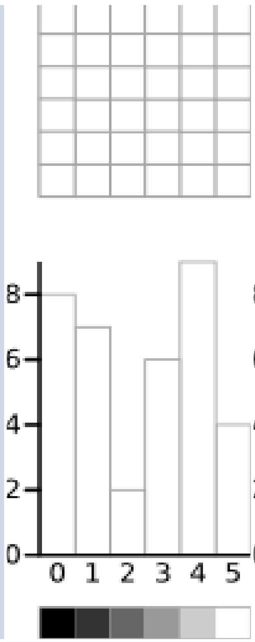
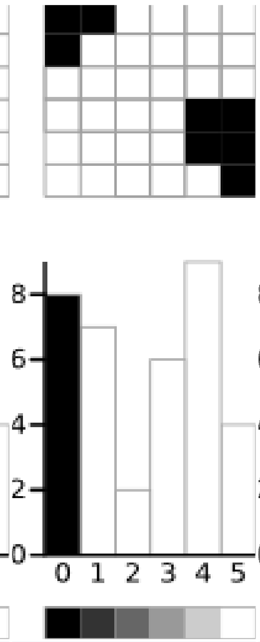
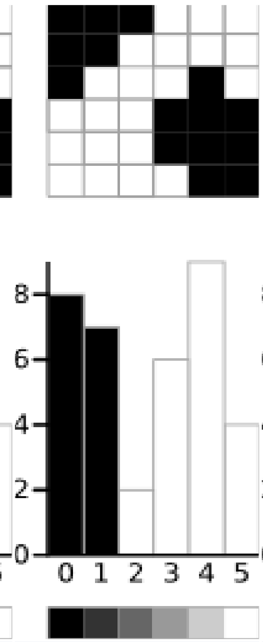
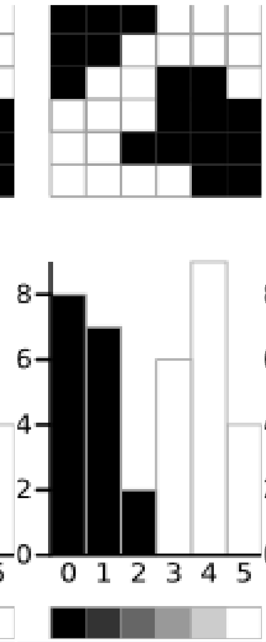
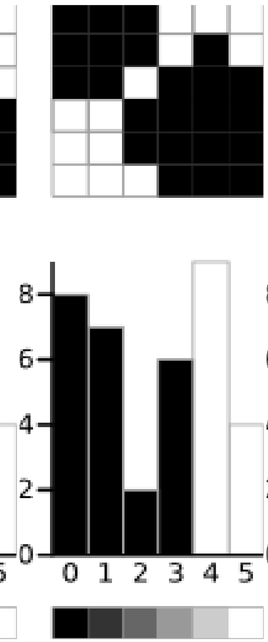
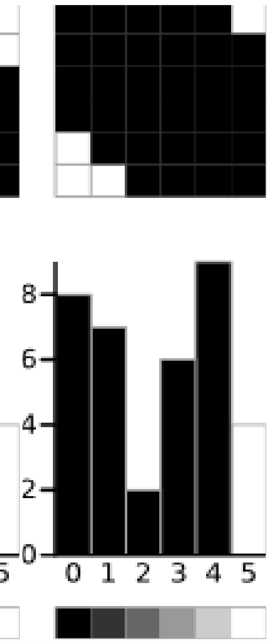
$$\mu_F = \frac{(3 \times 6) + (4 \times 9) + (5 \times 4)}{\mathrm{card}(F)} = 3.8947$$

$$\mu_{F^2} = \frac{(3^2 \times 6) + (4^2 \times 9) + (5^2 \times 4)}{\mathrm{card}(F)} = 15.6842$$

$$\sigma_F^2 = \mu_{F^2} - (\mu_F)^2 = 0.5152$$

On en déduit la variance intra-classe $\sigma_i^2 = \frac{\mathrm{card}(B)\times\sigma_B^2+\mathrm{card}(F)\times\sigma_F^2}{\mathrm{card}(B)+\mathrm{card}(F)} = 0.4909$

Filtre -- Seuillage – Otsu

Seuil	S=0	S=1	S=2	S=3	S=4	S=5
						
$\mathrm{card}(B)$	0	8	15	17	23	32
μ_B	0	0	0.4667	0.6471	1.2609	2.0313
σ_B^2	0	0	0.2489	0.4637	1.4102	2.5303
$\mathrm{card}(F)$	36	28	21	19	13	4
μ_F	2.3611	3.0357	3.7143	3.8947	4.3077	5.000
σ_F^2	3.1196	1.9639	0.7755	0.5152	0.2130	0
σ_i^2	3.1196	1.5268	0.5561	**0.4909**	0.9779	2.2491

Filtre

Seuillage – Otsu

- Méthode de calcul plus rapide :
  – Minimiser la variance intra-classe revient à maximiser la variance inter-classes

$$\sigma_o^2 = \sigma^2 - \sigma_i^2 = \frac{\mathrm{card}(F)\mathrm{card}(B)(\mu_B - \mu_F)^2}{\mathrm{card}(F) + \mathrm{card}(B)}$$

Seuil	S=0	S=1	S=2	S=3	S=4	S=5
σ_i^2	3.1196	1.5268	0.5561	0.4909	0.9779	2.2491
σ_o^2	0	1.5928	2.5635	2.6287	2.1417	0.8705

Filtre
Seuillage – σ-clipping

- Adapté pour chercher des objets brillant sur un fond sombre
- Hypothèse :
  - Le fond sombre est a peu prêt constant de moyenne μ
  - Le bruit est à peu prêt gaussien d'écart type σ
- Conséquence :
  - Plus de 99% des pixels dont la valeur est inférieur à $\mu+3\sigma$ font partis du fond
- Problème : comment déterminer μ et σ ?

Filtre

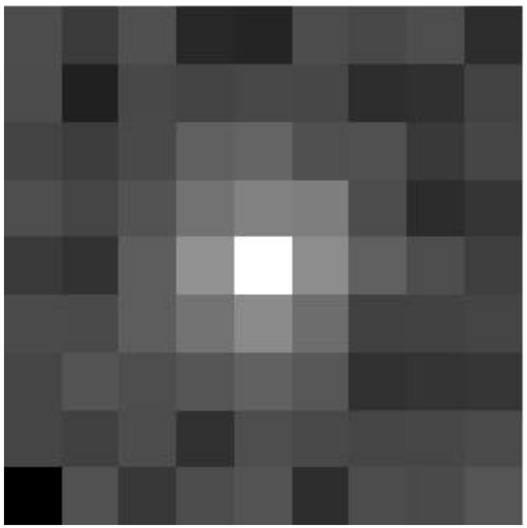
Seuillage – σ-clipping

- Procédure itérative avec masquage des pixels
- Répéter jusqu'à convergence
  1. Calculer μ et σ sur l'image
  2. Masquer les pixels dont la valeur est supérieure à $\mu + 3\sigma$

Filtre

Seuillage – σ-clipping

- Exemple :



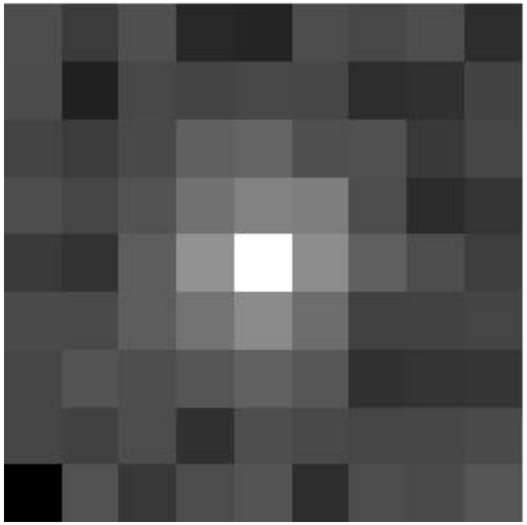
-2,26	1,35	-0,55	0,92	1,53	-1,12	0,96	0,66	1,74
0,37	0,03	0,95	-0,92	0,99	0,64	0,37	0,47	0,74
0,36	1,55	1,04	1,76	2,86	1,86	-0,92	-0,78	-0,68
0,77	0,66	2,52	5,08	8,89	4,30	-0,01	0,09	0,34
-0,45	-0,88	2,58	10,16	40,83	9,13	2,73	0,99	-0,09
1,00	0,29	1,49	5,00	7,35	6,70	0,90	-1,15	-0,74
0,24	-0,27	0,61	2,77	3,39	1,18	1,27	-0,47	0,41
0,88	-1,64	0,52	0,26	0,55	0,47	-1,15	-0,98	0,14
0,98	-0,39	1,23	-1,36	-1,48	0,91	0,53	1,03	-1,12

Itération 1 : $\mu = 1.6168$ $\sigma = 4.9995 \Rightarrow S = 16.6154$

Filtre

Seuillage – σ-clipping

Itération 1 : $\mu = 1.6168$ $\sigma = 4.9995 \Rightarrow S = 16.6154$

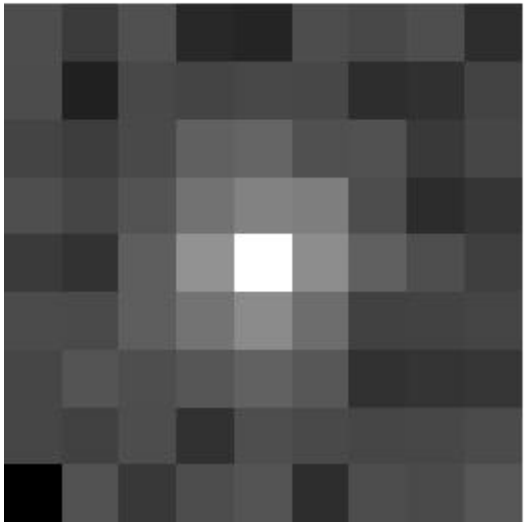


-2,26	1,35	-0,55	0,92	1,53	-1,12	0,96	0,66	1,74
0,37	0,03	0,95	-0,92	0,99	0,64	0,37	0,47	0,74
0,36	1,55	1,04	1,76	2,86	1,86	-0,92	-0,78	-0,68
0,77	0,66	2,52	5,08	8,89	4,30	-0,01	0,09	0,34
-0,45	-0,88	2,58	10,16	40,83	9,13	2,73	0,99	-0,09
1,00	0,29	1,49	5,00	7,35	6,70	0,90	-1,15	-0,74
0,24	-0,27	0,61	2,77	3,39	1,18	1,27	-0,47	0,41
0,88	-1,64	0,52	0,26	0,55	0,47	-1,15	-0,98	0,14
0,98	-0,39	1,23	-1,36	-1,48	0,91	0,53	1,03	-1,12

Itération 2 : $\mu = 1.1266$ $\sigma = 2.3665 \Rightarrow S = 8.2261$

Filtre
Seuillage – σ-clipping

Itération 2 : μ = 1.1266 σ = 2.3665 => S = 8.2261



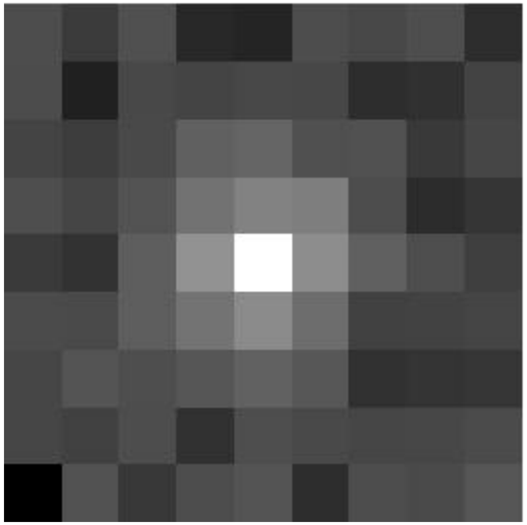
-2,26	1,35	-0,55	0,92	1,53	-1,12	0,96	0,66	1,74
0,37	0,03	0,95	-0,92	0,99	0,64	0,37	0,47	0,74
0,36	1,55	1,04	1,76	2,86	1,86	-0,92	-0,78	-0,68
0,77	0,66	2,52	5,08	8,89	4,30	-0,01	0,09	0,34
-0,45	-0,88	2,58	10,16	40,83	9,13	2,73	0,99	-0,09
1,00	0,29	1,49	5,00	7,35	6,70	0,90	-1,15	-0,74
0,24	-0,27	0,61	2,77	3,39	1,18	1,27	-0,47	0,41
0,88	-1,64	0,52	0,26	0,55	0,47	-1,15	-0,98	0,14
0,98	-0,39	1,23	-1,36	-1,48	0,91	0,53	1,03	-1,12

Itération 3 : μ = 0.8045 σ = 1.7338 => S = 6.0059

Filtre

Seuillage – σ-clipping

Itération 3 : $\mu = 0.8045$ $\sigma = 1.7338 \Rightarrow S = 6.0059$



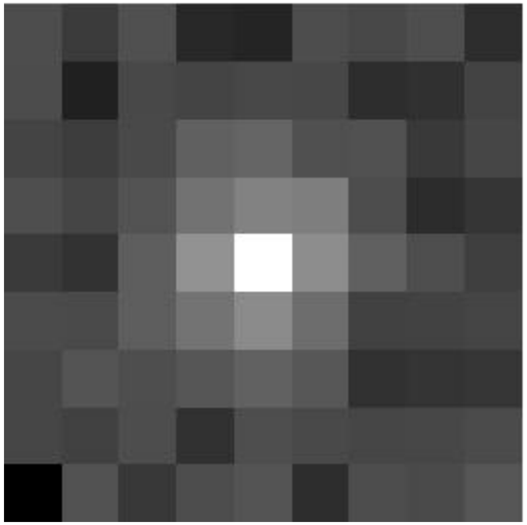
-2,26	1,35	-0,55	0,92	1,53	-1,12	0,96	0,66	1,74
0,37	0,03	0,95	-0,92	0,99	0,64	0,37	0,47	0,74
0,36	1,55	1,04	1,76	2,86	1,86	-0,92	-0,78	-0,68
0,77	0,66	2,52	5,08	8,89	4,30	-0,01	0,09	0,34
-0,45	-0,88	2,58	10,16	40,83	9,13	2,73	0,99	-0,09
1,00	0,29	1,49	5,00	7,35	6,70	0,90	-1,15	-0,74
0,24	-0,27	0,61	2,77	3,39	1,18	1,27	-0,47	0,41
0,88	-1,64	0,52	0,26	0,55	0,47	-1,15	-0,98	0,14
0,98	-0,39	1,23	-1,36	-1,48	0,91	0,53	1,03	-1,12

Itération 4 : $\mu = 0.6385$ $\sigma = 1.4179 \Rightarrow S = 4.8921$

Filtre

Seuillage – σ-clipping

Itération 4 : $\mu = 0.6385$ $\sigma = 1.4179 \Rightarrow S = 4.8921$



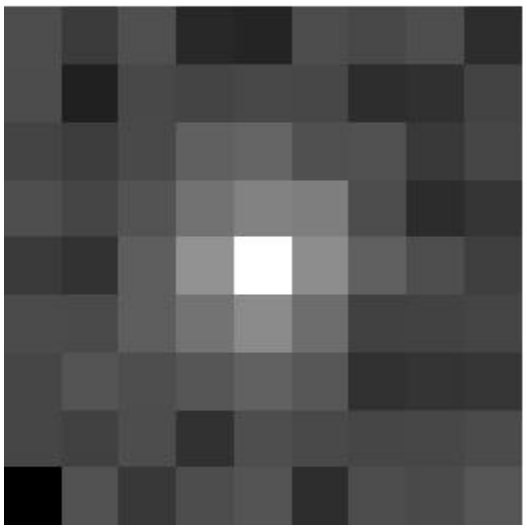
-2,26	1,35	-0,55	0,92	1,53	-1,12	0,96	0,66	1,74
0,37	0,03	0,95	-0,92	0,99	0,64	0,37	0,47	0,74
0,36	1,55	1,04	1,76	2,86	1,86	-0,92	-0,78	-0,68
0,77	0,66	2,52	5,08	8,89	4,30	-0,01	0,09	0,34
-0,45	-0,88	2,58	10,16	40,83	9,13	2,73	0,99	-0,09
1,00	0,29	1,49	5,00	7,35	6,70	0,90	-1,15	-0,74
0,24	-0,27	0,61	2,77	3,39	1,18	1,27	-0,47	0,41
0,88	-1,64	0,52	0,26	0,55	0,47	-1,15	-0,98	0,14
0,98	-0,39	1,23	-1,36	-1,48	0,91	0,53	1,03	-1,12

Itération 5 : $\mu = 0.5179$ $\sigma = 1.2298 \Rightarrow S = 4.2072$

Filtre

Seuillage – σ-clipping

Itération 5 : $\mu = 0.5179$ $\sigma = 1.2298 \Rightarrow S = 4.2072$

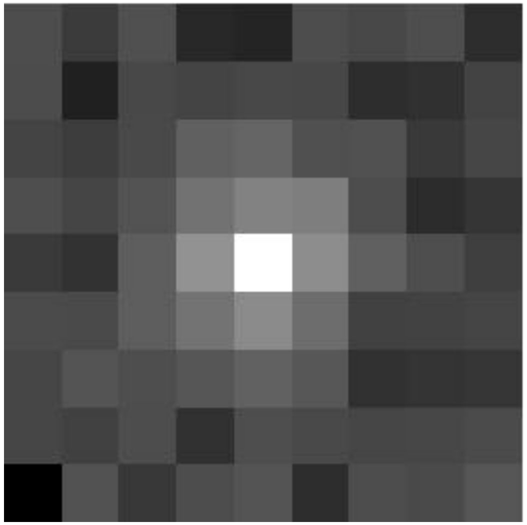


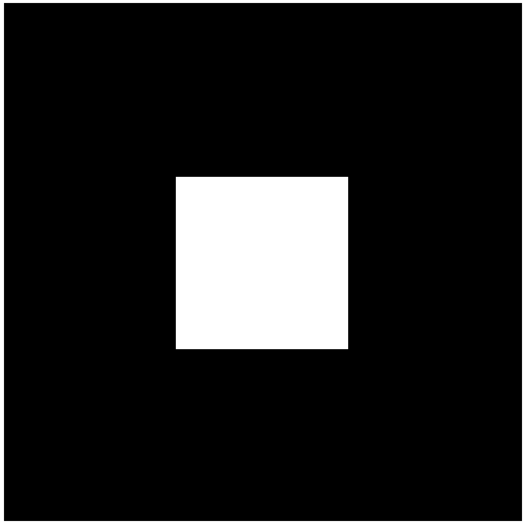
-2,26	1,35	-0,55	0,92	1,53	-1,12	0,96	0,66	1,74
0,37	0,03	0,95	-0,92	0,99	0,64	0,37	0,47	0,74
0,36	1,55	1,04	1,76	2,86	1,86	-0,92	-0,78	-0,68
0,77	0,66	2,52	5,08	8,89	4,30	-0,01	0,09	0,34
-0,45	-0,88	2,58	10,16	40,83	9,13	2,73	0,99	-0,09
1,00	0,29	1,49	5,00	7,35	6,70	0,90	-1,15	-0,74
0,24	-0,27	0,61	2,77	3,39	1,18	1,27	-0,47	0,41
0,88	-1,64	0,52	0,26	0,55	0,47	-1,15	-0,98	0,14
0,98	-0,39	1,23	-1,36	-1,48	0,91	0,53	1,03	-1,12

Itération 6 : $\mu = 0.4654$ $\sigma = 1.1531 \Rightarrow S = 3.9246$

Filtre

Seuillage – σ-clipping





Itération 6 : $\mu = 0.4654$ $\sigma = 1.1531 \Rightarrow S = 3.9246$

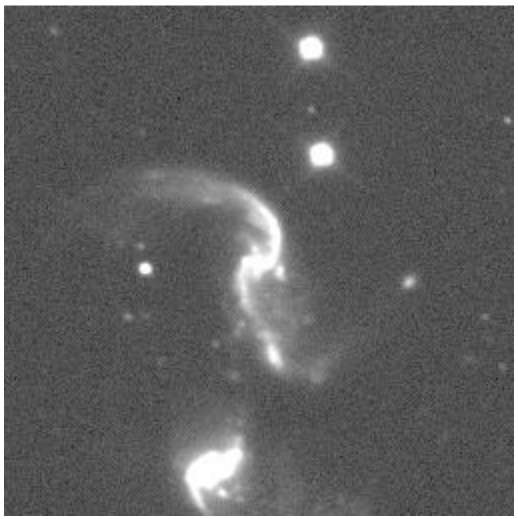
Plus de changement !

-2,26	1,35	-0,55	0,92	1,53	-1,12	0,96	0,66	1,74
0,37	0,03	0,95	-0,92	0,99	0,64	0,37	0,47	0,74
0,36	1,55	1,04	1,76	2,86	1,86	-0,92	-0,78	-0,68
0,77	0,66	2,52	5,08	8,89	4,30	-0,01	0,09	0,34
-0,45	-0,88	2,58	10,16	40,83	9,13	2,73	0,99	-0,09
1,00	0,29	1,49	5,00	7,35	6,70	0,90	-1,15	-0,74
0,24	-0,27	0,61	2,77	3,39	1,18	1,27	-0,47	0,41
0,88	-1,64	0,52	0,26	0,55	0,47	-1,15	-0,98	0,14
0,98	-0,39	1,23	-1,36	-1,48	0,91	0,53	1,03	-1,12

Filtre

Seuillage – σ-clipping

- Démo sur image astronomique



Originale

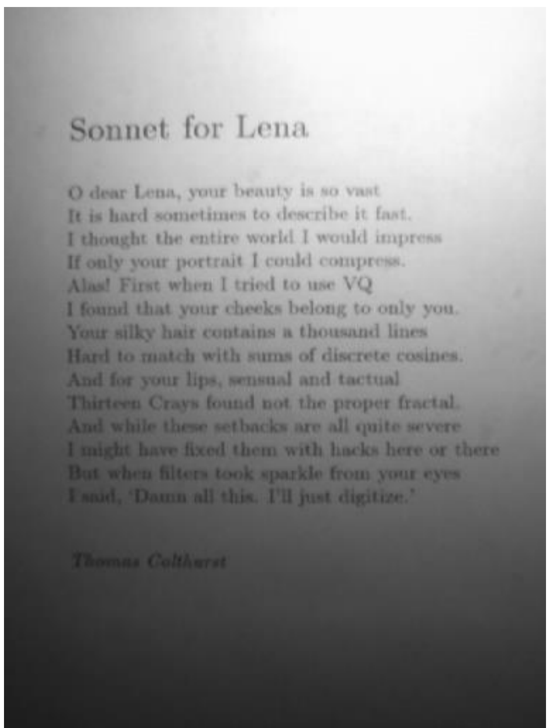
Seuillage avec sigma-clipping

Filtrage morphologique open-close avec une boule de rayon 2 pixels

Filtre

Seuillage – Approche locale

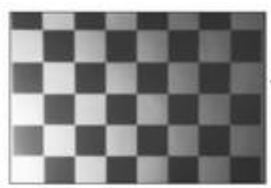
- Les méthodes présentées précédemment font toutes l'hypothèse que les conditions d'illumination ne varient pas dans l'image
- Ce n'est généralement pas vrai !

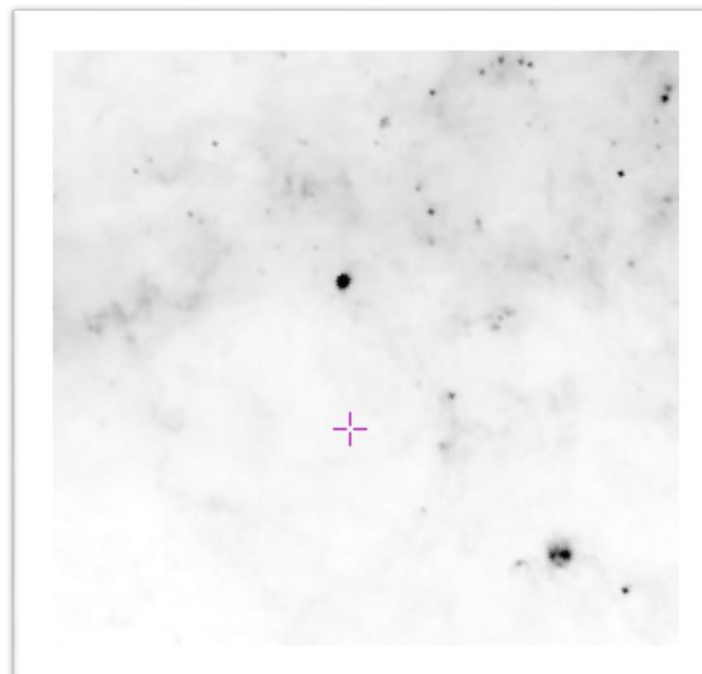


Sonnet for Lena

O dear Lena, your beauty is so vast
It is hard sometimes to describe it fast.
I thought the entire world I would impress
If only your portrait I could compress.
Alas! First when I tried to use VQ
I found that your cheeks belong to only you.
Your silky hair contains a thousand lines
Hard to match with sums of discrete cosines.
And for your lips, sensual and tactual
Thirteen Crays found not the proper fractal.
And while these setbacks are all quite severe
I might have fixed them with hacks here or there
But when filters took sparkle from your eyes
I said, 'Damn all this. I'll just digitize.'

Thomas Colthurst





Filtre

Seuillage – Approche locale

- La solution est simple : au lieu de procéder de manière globale on va travailler sur des fenêtres glissantes

Méthode de seuillage traditionnelle (otsu, sigma clipping...) appliquée dans une fenêtre qui se déplace sur l'image