Traitement d'images

Emanuel Aldea <emanuel.aldea@u-psud.fr>
http://hebergement.u-psud.fr/emi/TIPolytech

Polytech Paris-Saclay 5$^{\text{ère}}$ année

Organisation du cours

Contenu des enseignements TI :

- Cours : 8h, TD : 8h, TP : 8h
- Support de cours/TD en ligne

Séance	Cours & TD	TP
S1	4h00	-
S2-S3	2h00	2h00
S4	4h00	-
S5-S6	2h00	2h00

Organisation du cours

Contenu des enseignements TI :

- Cours : 8h, TD : 8h, TP : 8h
- Support de cours/TD en ligne

Séance	Cours & TD	TP
S1	4h00	-
S2-S3	2h00	2h00
S4	4h00	-
S5-S6	2h00	2h00

Modalités d'évaluation :

- A définir avec vous

Organisation du cours

Contenu des enseignements TI :

- Cours : 8h, TD : 8h, TP : 8h
- Support de cours/TD en ligne

Séance	Cours & TD	TP
S1	4h00	-
S2-S3	2h00	2h00
S4	4h00	-
S5-S6	2h00	2h00

Modalités d'évaluation :

- A définir avec vous

Contrôle continu TI :

- A définir avec vous

Organisation du cours

Séance	Cours & TD	TP
S1	4h00	-
S2-S3	2h00	2h00
S4	4h00	-
S5-S6	2h00	2h00

Contenu des enseignements TI :

- Cours : 8h, TD : 8h, TP : 8h
- Support de cours/TD en ligne

Modalités d'évaluation :

- A définir avec vous

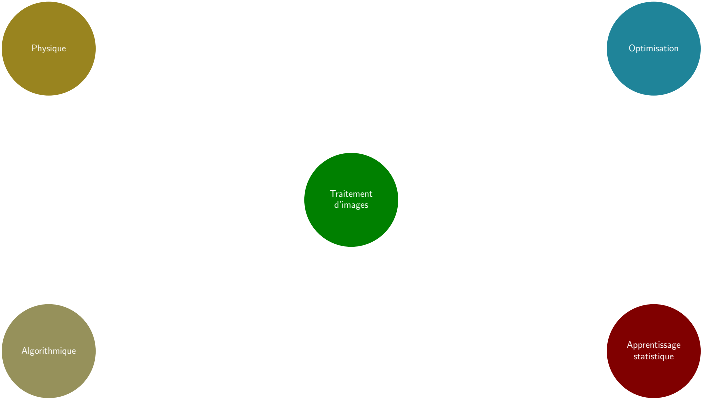
Contrôle continu TI :

- A définir avec vous

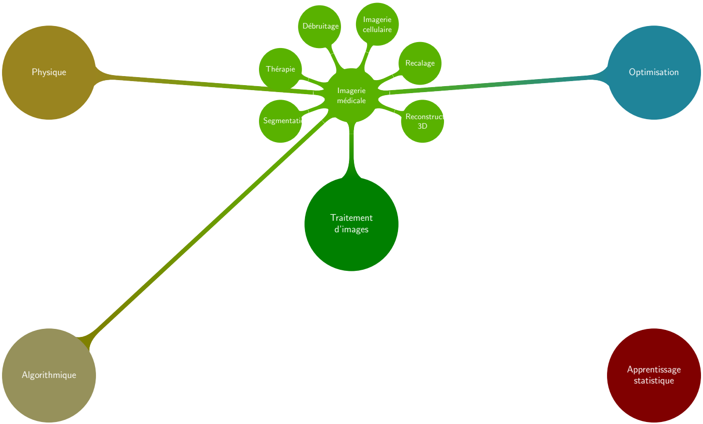
Pour vous connaître mieux :

- Familiarité avec Matlab[1-5], C++[1-5]
- Compréhension [1-5]
- Intéressant [1-5]
- Remarques personnelles :
  - objectif particulier lié au TI (ou pas !)
  - exemple de projet TI qui vous passionnerait
  - etc.

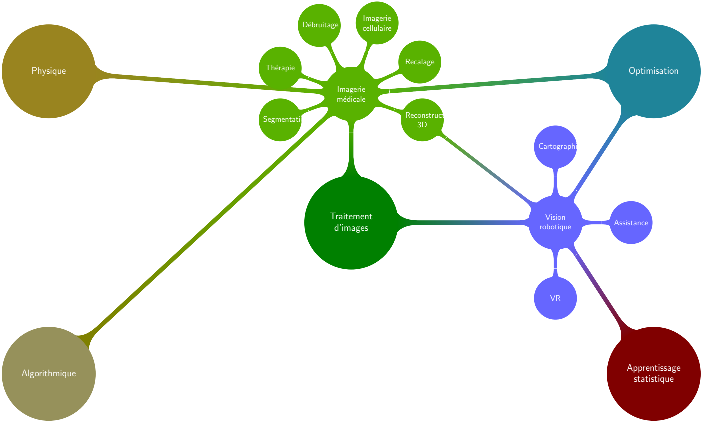
Domaines d'application (quelques exemples)



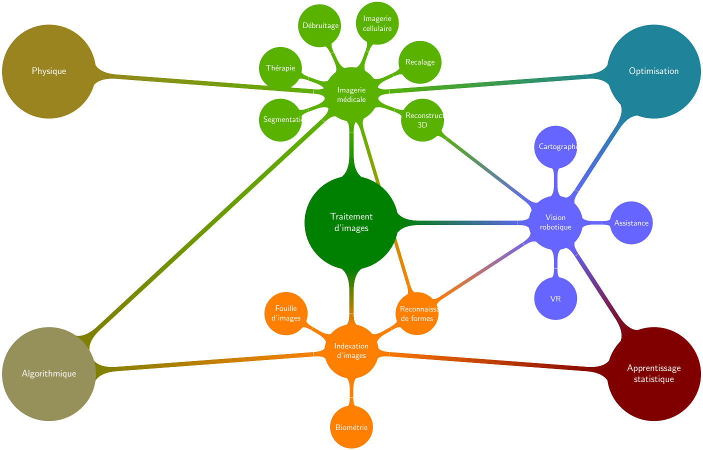
Domaines d'application (quelques exemples)



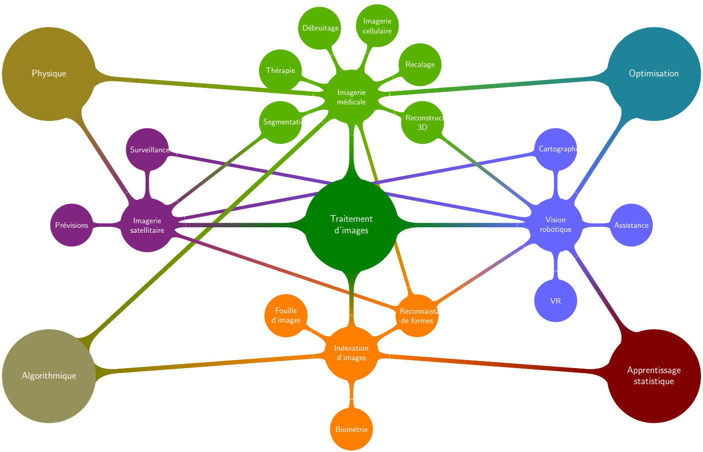
Domaines d'application (quelques exemples)



Domaines d'application (quelques exemples)

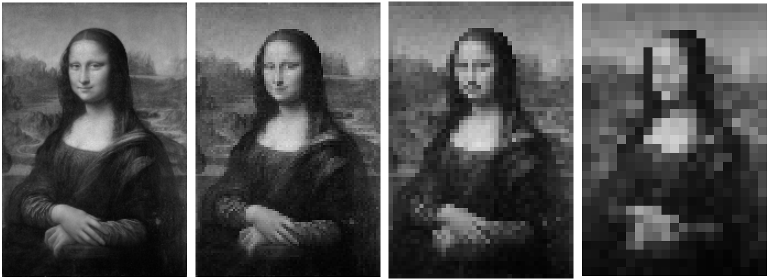


Domaines d'application (quelques exemples)



(Une) Définition

Image : représentation *continue* d'une fonction $f(x, y)$ qui relie f à l'intensité lumineuse du point (x, y)
Image numérique : échantillonnage $I(x, y)$ discret (matrice 2D) de f qui relie $I(x, y)$ à l'intensité lumineuse d'une case (x, y), nommée **pixel**



168x250 84x125 42x63 21x31

FIGURE – Échantillonnage (discrétisation **spatiale**)

(Une) Définition

Image : représentation *continue* d'une fonction $f(x, y)$ qui relie f à l'intensité lumineuse du point (x, y)

Image numérique : échantillonnage $I(x, y)$ discret (matrice 2D) de f qui relie $I(x, y)$ à l'intensité lumineuse d'une case (x, y), nommée **pixel**

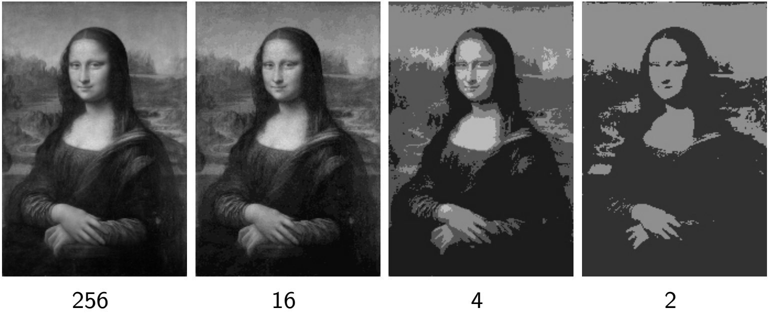
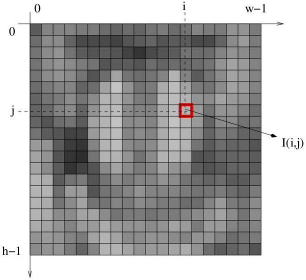


FIGURE – Quantification (discrétisation **tonale)**

Notations et structure

Accès aux pixels

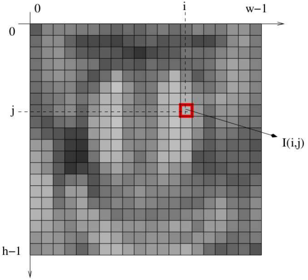
- w : nombre de colonnes, index $i \in [0, w-1]$
- h : nombre de lignes, index $j \in [0, h-1]$
- $I(i,j)$: valeur pixel $i^{\text{ème}}$ colonne et $j^{\text{ème}}$ ligne



Notations et structure

Accès aux pixels

- w : nombre de colonnes, index $i \in [0, w-1]$
- h : nombre de lignes, index $j \in [0, h-1]$
- $I(i,j)$: valeur pixel $i^{\text{ème}}$ colonne et $j^{\text{ème}}$ ligne



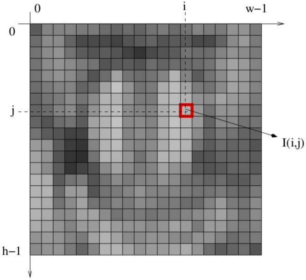
Valeur des pixels (cas habituels)

- gris : intensité comme scalaire en $[0, 2^n - 1]$
  - n : dynamique de l'image
  - $N = 2^n$: nombre de niveaux de gris
  - $n = 8$ habituellement

Notations et structure

Accès aux pixels

- w : nombre de colonnes, index $i \in [0, w-1]$
- h : nombre de lignes, index $j \in [0, h-1]$
- $I(i,j)$: valeur pixel $i^{\text{ème}}$ colonne et $j^{\text{ème}}$ ligne



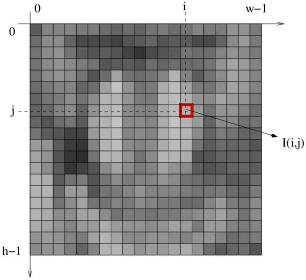
Valeur des pixels (cas habituels)

- gris : intensité comme scalaire en $[0, 2^n - 1]$
  - n : dynamique de l'image
  - $N = 2^n$: nombre de niveaux de gris
  - $n = 8$ habituellement
- couleur : triplet correspondant aux intensités des canaux R, G et B
  - chaque canal encodé comme précédemment

Notations et structure

Accès aux pixels

- w : nombre de colonnes, index $i \in [0, w-1]$
- h : nombre de lignes, index $j \in [0, h-1]$
- $I(i,j)$: valeur pixel $i^{\text{ème}}$ colonne et $j^{\text{ème}}$ ligne



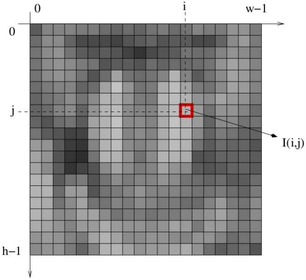
Valeur des pixels (cas habituels)

- gris : intensité comme scalaire en $[0, 2^n - 1]$
  - n : dynamique de l'image
  - $N = 2^n$: nombre de niveaux de gris
  - $n = 8$ habituellement
- couleur : triplet correspondant aux intensités des canaux R, G et B
  - chaque canal encodé comme précédemment

Notations et structure

Accès aux pixels

- w : nombre de colonnes, index $i \in [0, w-1]$
- h : nombre de lignes, index $j \in [0, h-1]$
- $I(i,j)$: valeur pixel $i^{\text{ème}}$ colonne et $j^{\text{ème}}$ ligne



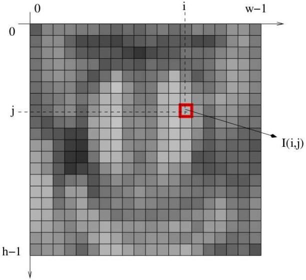
Valeur des pixels (cas habituels)

- gris : intensité comme scalaire en $[0, 2^n - 1]$
  - n : dynamique de l'image
  - $N = 2^n$: nombre de niveaux de gris
  - $n = 8$ habituellement
- couleur : triplet correspondant aux intensités des canaux R, G et B
  - chaque canal encodé comme précédemment

Notations et structure

Accès aux pixels

- w : nombre de colonnes, index $i \in [0, w-1]$
- h : nombre de lignes, index $j \in [0, h-1]$
- $I(i,j)$: valeur pixel $i^{\text{ème}}$ colonne et $j^{\text{ème}}$ ligne



Valeur des pixels (cas habituels)

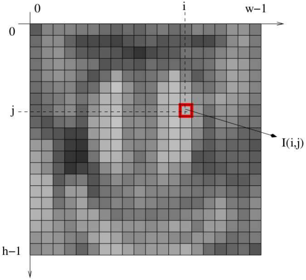
- gris : intensité comme scalaire en $[0, 2^n - 1]$
  - n : dynamique de l'image
  - $N = 2^n$: nombre de niveaux de gris
  - $n = 8$ habituellement
- couleur : triplet correspondant aux intensités des canaux R, G et B
  - chaque canal encodé comme précédemment

bleu

Notations et structure

Accès aux pixels

- w : nombre de colonnes, index $i \in [0, w-1]$
- h : nombre de lignes, index $j \in [0, h-1]$
- $I(i,j)$: valeur pixel $i^{\text{ème}}$ colonne et $j^{\text{ème}}$ ligne



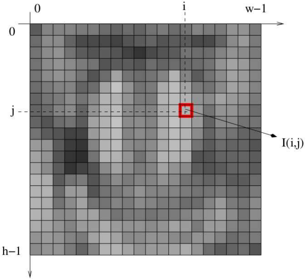
Valeur des pixels (cas habituels)

- gris : intensité comme scalaire en $[0, 2^n - 1]$
  - n : dynamique de l'image
  - $N = 2^n$: nombre de niveaux de gris
  - $n = 8$ habituellement
- couleur : triplet correspondant aux intensités des canaux R, G et B
  - chaque canal encodé comme précédemment

Notations et structure

Accès aux pixels

- w : nombre de colonnes, index $i \in [0, w-1]$
- h : nombre de lignes, index $j \in [0, h-1]$
- $I(i,j)$: valeur pixel $i^{\text{ème}}$ colonne et $j^{\text{ème}}$ ligne



Valeur des pixels (cas habituels)

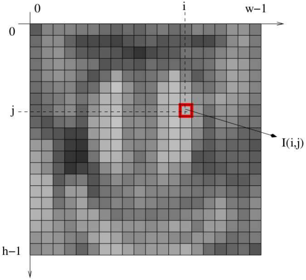
- gris : intensité comme scalaire en $[0, 2^n - 1]$
  - n : dynamique de l'image
  - $N = 2^n$: nombre de niveaux de gris
  - $n = 8$ habituellement
- couleur : triplet correspondant aux intensités des canaux R, G et B
  - chaque canal encodé comme précédemment

rouge

Notations et structure

Accès aux pixels

- w : nombre de colonnes, index $i \in [0, w-1]$
- h : nombre de lignes, index $j \in [0, h-1]$
- $I(i,j)$: valeur pixel $i^{\text{ème}}$ colonne et $j^{\text{ème}}$ ligne



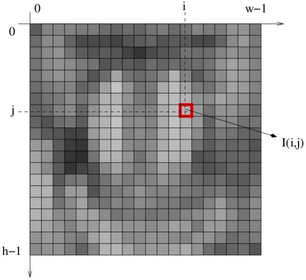
Valeur des pixels (cas habituels)

- gris : intensité comme scalaire en $[0, 2^n - 1]$
  - n : dynamique de l'image
  - $N = 2^n$: nombre de niveaux de gris
  - $n = 8$ habituellement
- couleur : triplet correspondant aux intensités des canaux R, G et B
  - chaque canal encodé comme précédemment

Notations et structure

Accès aux pixels

- w : nombre de colonnes, index $i \in [0, w-1]$
- h : nombre de lignes, index $j \in [0, h-1]$
- $I(i,j)$: valeur pixel $i^{\text{ème}}$ colonne et $j^{\text{ème}}$ ligne



Valeur des pixels (cas habituels)

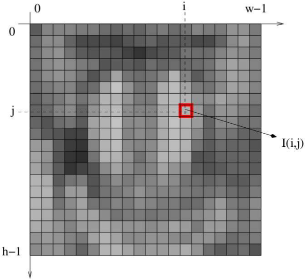
- gris : intensité comme scalaire en $[0, 2^n - 1]$
  - n : dynamique de l'image
  - $N = 2^n$: nombre de niveaux de gris
  - $n = 8$ habituellement
- couleur : triplet correspondant aux intensités des canaux R, G et B
  - chaque canal encodé comme précédemment

vert

Notations et structure

Accès aux pixels

- w : nombre de colonnes, index $i \in [0, w-1]$
- h : nombre de lignes, index $j \in [0, h-1]$
- $I(i,j)$: valeur pixel $i^{\text{ème}}$ colonne et $j^{\text{ème}}$ ligne



Valeur des pixels (cas habituels)

- gris : intensité comme scalaire en $[0, 2^n - 1]$
  - n : dynamique de l'image
  - $N = 2^n$: nombre de niveaux de gris
  - $n = 8$ habituellement
- couleur : triplet correspondant aux intensités des canaux R, G et B
  - chaque canal encodé comme précédemment



Amélioration d'images

Objectifs

- Comment rehausser le contraste d'une image de façon à faire apparaître les objets ?

Amélioration d'images

Objectifs

- Comment rehausser le contraste d'une image de façon à faire apparaître les objets ?
- Comment s'affranchir des paramètres de luminosité lors de l'acquisition ?
  - Exemple : étalonnage des intensités en vue de leur comparaison

Amélioration d’images

Objectifs

- Comment rehausser le contraste d’une image de façon à faire apparaître les objets ?
- Comment s’affranchir des paramètres de luminosité lors de l’acquisition ?
  - Exemple : étalonnage des intensités en vue de leur comparaison
  - Application : mise en correspondance, détection de changement, classification etc

Amélioration d'images

Objectifs

- Comment rehausser le contraste d'une image de façon à faire apparaître les objets ?
- Comment s'affranchir des paramètres de luminosité lors de l'acquisition ?
  - Exemple : étalonnage des intensités en vue de leur comparaison
  - Application : mise en correspondance, détection de changement, classification etc
- Histogramme : modèle probabiliste empirique
- Généralement appliquée aux images en niveaux de gris

L'histogramme de l'image

Définition

- Résultat de la quantification (N niveaux de gris possibles)
- Histogramme $H : [0, N-1] \rightarrow [0, wh]$:
  - $H(z) = card(\{(x,y) \in [0,w] \times [0,h] | I(x,y) = z\})$

L'histogramme de l'image

Définition

- Résultat de la quantification (N niveaux de gris possibles)
- Histogramme $H : [0, N-1] \rightarrow [0, wh]$:
  - $H(z) = card(\{(x, y) \in [0, w] \times [0, h] | I(x, y) = z\})$
- Histogramme normalisé $H_n : [0, N-1] \rightarrow [0, 1]$:
  - $H_n(z) = H(z)/(wh)$
  - distribution de probabilité empirique

L'histogramme de l'image

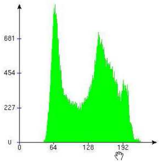
Définition

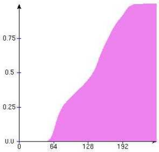
- Résultat de la quantification (N niveaux de gris possibles)
- Histogramme $H : [0, N-1] \rightarrow [0, wh]$:
  - $H(z) = card(\{(x,y) \in [0,w] \times [0,h] | I(x,y) = z\})$
- Histogramme normalisé $H_n : [0, N-1] \rightarrow [0, 1]$:
  - $H_n(z) = H(z)/(wh)$
  - distribution de probabilité empirique
- Histogramme cumulé normalisé $H_{cn} : [0, N-1] \rightarrow [0, 1]$:
  - $H_{cn}(z) = \sum_0^z H_n(z)$
  - fonction de répartition empirique
  - fonction croissante

L'histogramme de l'image

Définition

- Résultat de la quantification (N niveaux de gris possibles)
- Histogramme $H : [0, N-1] \rightarrow [0, wh]$:
  - $H(z) = card(\{(x, y) \in [0, w] \times [0, h] | I(x, y) = z\})$
- Histogramme normalisé $H_n : [0, N-1] \rightarrow [0, 1]$:
  - $H_n(z) = H(z)/(wh)$
  - distribution de probabilité empirique
- Histogramme cumulé normalisé $H_{cn} : [0, N-1] \rightarrow [0, 1]$:
  - $H_{cn}(z) = \sum_0^z H_n(z)$
  - fonction de répartition empirique
  - fonction croissante





L'histogramme de l'image

Calcul des histogrammes

```
int H[N]; // histogramme
float Hn[N]; // histogramme cumulé
float Hcn[N]; // histogramme cumulé normalisé
for (i = 0; i<N; i++){
  H[i] = 0;
  Hn[i] = 0;
  Hc[i] = 0;
}
// calcul de H
```

L'histogramme de l'image

Calcul des histogrammes

```
int H[N]; // histogramme
float Hn[N]; // histogramme cumulé
float Hcn[N]; // histogramme cumulé normalisé
for (i = 0; i<N; i++){
  H[i] = 0;
  Hn[i] = 0;
  Hc[i] = 0;
}
// calcul de H
for (i = 0; i<w; i++)
  for (j = 0; j<h; j++){
    int val = I(i,j);
    H[val] = H[val] + 1;
  }
// calcul de Hn,Hcn
```

L'histogramme de l'image

Calcul des histogrammes

```
int H[N]; // histogramme
float Hn[N]; // histogramme cumulé
float Hcn[N]; // histogramme cumulé normalisé
for (i = 0; i<N; i++){
  H[i] = 0;
  Hn[i] = 0;
  Hc[i] = 0;
}
// calcul de H
for (i = 0; i<w; i++)
  for (j = 0; j<h; j++){
    int val = I(i,j);
    H[val] = H[val] + 1;
  }
// calcul de Hn,Hcn


Hc[0] = Hn[0] = H[0] / (w*h);
for (i = 1; i<N; i++){
  Hn[i] = H[i] / (w*h);
  Hcn[i] = Hcn[i-1] + Hn[i];
}
```

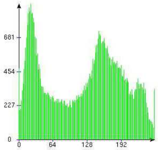
Transformations de l'histogramme

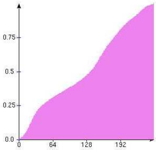
Principe

- Ne pas altérer la relation d'ordre
- Étalement de la dynamique
  - transformation **linéaire** de $z \in [z_{min}, z_{max}]$ vers $z' \in [z'_{min}, z'_{max}]$:

$$z' = z'_{min} + (z - z_{min}) \frac{z'_{max} - z'_{min}}{z_{max} - z_{min}}$$

  - suite à la quantification $z' = round(z')$
  - généralement $z'_{min} = 0, z'_{max} = 255$
  - souvent peu pratique car sensible au bruit



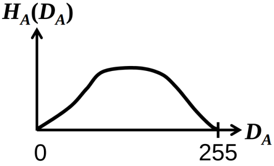


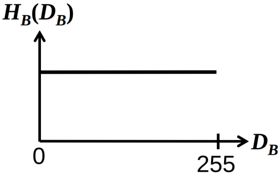
Transformations de l'histogramme

Égalisation d'histogramme

- Ne pas altérer la relation d'ordre
- S'approcher d'un H hétérogène /plat par une transformation f **non-linéaire**
- Ou autrement dit d'un H_{cn} uniformément croissant

$$D_B = f(D_A) = \frac{D_{max}}{S} \int_0^{D_A} H_A(u)du$$



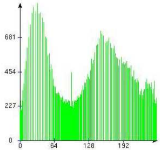


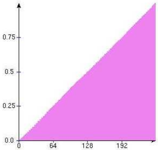
Transformations de l'histogramme

Égalisation d'histogramme

- Ne pas altérer la relation d'ordre
- S'approcher d'un H hétérogène /plat
- Ou autrement dit d'un H_{cn} uniformément croissant

$$z' = \frac{N-1}{wh} \sum_{i=0}^{z} H(i) = (N-1) H_{cn}(z)$$





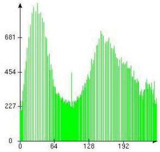
Transformations de l'histogramme

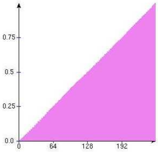
Égalisation d'histogramme

- Ne pas altérer la relation d'ordre
- S'approcher d'un H hétérogène /plat
- Ou autrement dit d'un H_{cn} uniformément croissant

$$z' = \frac{N-1}{wh} \sum_{i=0}^{z} H(i) = (N-1) H_{cn}(z)$$

- Suite à la quantification $z' = round(z')$





Objectifs

- Amélioration
  - Comment réduire le bruit d'une image de façon à améliorer la "netteté" des objets ?
  - Prétraitements
  - Visualisation
- Simplification
  - Comment réduire la variabilité intrinsèque des objets de façon à les simplifier ?
  - Hypothèse : profil spécifique pour l'information utile
  - Applications : analyse statistique, traitement automatique, classification

Bruit

Apparition

- erreurs générant dans les pixels des valeurs différentes des valeurs réelles
- indépendant en chaque pixel
- sources : capteur, transmission, interférences etc.
- additif, multiplicatif, impulsionnel etc.
- additif : $I(i,j) = I_0(i,j) + \eta(i,j)$

Bruit

Apparition

- erreurs générant dans les pixels des valeurs différentes des valeurs réelles
- indépendant en chaque pixel
- sources : capteur, transmission, interférences etc.
- additif, multiplicatif, impulsionnel etc.
- additif : $I(i,j) = I_0(i,j) + \eta(i,j)$

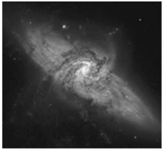
Bruit Gaussien

- bon modèle pour bruit capteurs : $p(\eta) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{(\eta-\mu)^2}{2\sigma^2})$

Bruit

Apparition

- erreurs générant dans les pixels des valeurs différentes des valeurs réelles
- indépendant en chaque pixel
- sources : capteur, transmission, interférences etc.
- additif, multiplicatif, impulsionnel etc.
- additif : $I(i,j) = I_0(i,j) + \eta(i,j)$



référence

Bruit Gaussien

- bon modèle pour bruit capteurs : $p(\eta) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{(\eta-\mu)^2}{2\sigma^2})$
- exemple : moyenne sur plusieurs acquisitions (astronomie)

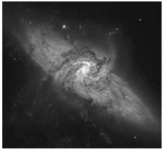
Bruit

Apparition

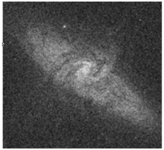
- erreurs générant dans les pixels des valeurs différentes des valeurs réelles
- indépendant en chaque pixel
- sources : capteur, transmission, interférences etc.
- additif, multiplicatif, impulsionnel etc.
- additif : $I(i,j) = I_0(i,j) + \eta(i,j)$

Bruit Gaussien

- bon modèle pour bruit capteurs : $p(\eta) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{(\eta-\mu)^2}{2\sigma^2})$
- exemple : moyenne sur plusieurs acquisitions (astronomie)



référence



$\sigma = 64$

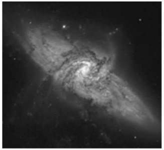
Bruit

Apparition

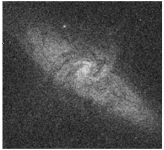
- erreurs générant dans les pixels des valeurs différentes des valeurs réelles
- indépendant en chaque pixel
- sources : capteur, transmission, interférences etc.
- additif, multiplicatif, impulsionnel etc.
- additif : $I(i,j) = I_0(i,j) + \eta(i,j)$

Bruit Gaussien

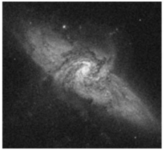
- bon modèle pour bruit capteurs : $p(\eta) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{(\eta-\mu)^2}{2\sigma^2})$
- exemple : moyenne sur plusieurs acquisitions (astronomie)



référence



$\sigma = 64$



$K = 8$

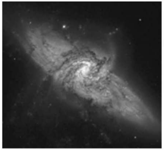
Bruit

Apparition

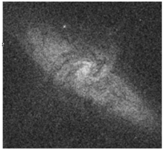
- erreurs générant dans les pixels des valeurs différentes des valeurs réelles
- indépendant en chaque pixel
- sources : capteur, transmission, interférences etc.
- additif, multiplicatif, impulsionnel etc.
- additif : $I(i,j) = I_0(i,j) + \eta(i,j)$

Bruit Gaussien

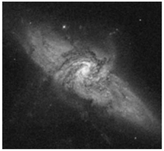
- bon modèle pour bruit capteurs : $p(\eta) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{(\eta-\mu)^2}{2\sigma^2})$
- exemple : moyenne sur plusieurs acquisitions (astronomie)



référence



$\sigma = 64$

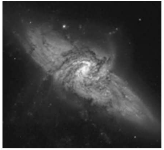


$K = 16$

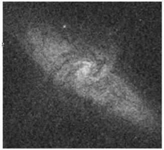
Bruit

Apparition

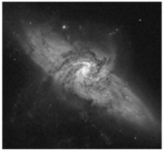
- erreurs générant dans les pixels des valeurs différentes des valeurs réelles
- indépendant en chaque pixel
- sources : capteur, transmission, interférences etc.
- additif, multiplicatif, impulsionnel etc.
- additif : $I(i,j) = I_0(i,j) + \eta(i,j)$



référence



$\sigma = 64$



$K = 64$

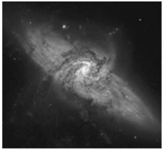
Bruit Gaussien

- bon modèle pour bruit capteurs : $p(\eta) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{(\eta-\mu)^2}{2\sigma^2})$
- exemple : moyenne sur plusieurs acquisitions (astronomie)

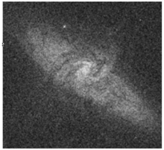
Bruit

Apparition

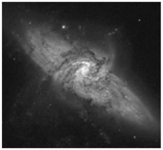
- erreurs générant dans les pixels des valeurs différentes des valeurs réelles
- indépendant en chaque pixel
- sources : capteur, transmission, interférences etc.
- additif, multiplicatif, impulsionnel etc.
- additif : $I(i,j) = I_0(i,j) + \eta(i,j)$



référence



$\sigma = 64$



$K = 128$

Bruit Gaussien

- bon modèle pour bruit capteurs : $p(\eta) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{(\eta-\mu)^2}{2\sigma^2})$
- exemple : moyenne sur plusieurs acquisitions (astronomie)

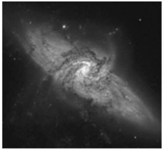
Bruit

Apparition

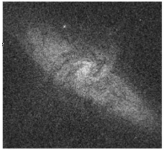
- erreurs générant dans les pixels des valeurs différentes des valeurs réelles
- indépendant en chaque pixel
- sources : capteur, transmission, interférences etc.
- additif, multiplicatif, impulsionnel etc.
- additif : $I(i,j) = I_0(i,j) + \eta(i,j)$

Bruit Gaussien

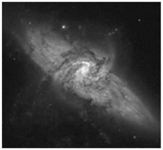
- bon modèle pour bruit capteurs : $p(\eta) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{(\eta-\mu)^2}{2\sigma^2})$
- exemple : moyenne sur plusieurs acquisitions (astronomie)



référence



$\sigma = 64$



$K = 128$

Exercice :

Quel est le σ pour la dernière image ($K = 128$) ?

Bruit

Apparition

- erreurs générant dans les pixels des valeurs différentes des valeurs réelles
- indépendant en chaque pixel
- sources : capteur, transmission, interférences etc.
- additif, multiplicatif, impulsionnel etc.
- additif : $I(i,j) = I_0(i,j) + \eta(i,j)$

Bruit

Apparition

- erreurs générant dans les pixels des valeurs différentes des valeurs réelles
- indépendant en chaque pixel
- sources : capteur, transmission, interférences etc.
- additif, multiplicatif, impulsionnel etc.
- additif : $I(i,j) = I_0(i,j) + \eta(i,j)$

référence

B. impulsionnel

Bruit impulsionnel

- "poivre et sel", ou salt-and-pepper
- conversion, transmission, pixels "morts"
- certaines valeurs très différentes en intensité

Bruit

Apparition

- erreurs générant dans les pixels des valeurs différentes des valeurs réelles
- indépendant en chaque pixel
- sources : capteur, transmission, interférences etc.
- additif, multiplicatif, impulsionnel etc.
- additif : $I(i,j) = I_0(i,j) + \eta(i,j)$

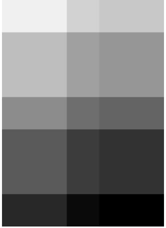
Bruit

Apparition

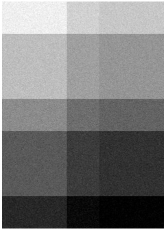
- erreurs générant dans les pixels des valeurs différentes des valeurs réelles
- indépendant en chaque pixel
- sources : capteur, transmission, interférences etc.
- additif, multiplicatif, impulsionnel etc.
- additif : $I(i,j) = I_0(i,j) + \eta(i,j)$

Bruit multiplicatif

- images radar, laser
- effets photochimiques (bruit “grain”)
- $I(i,j) = I_0(i,j) \cdot \eta(i,j), E[\eta] = 1$



référence

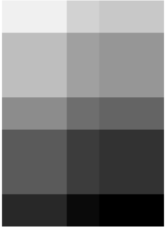


B. additif (Var=60)

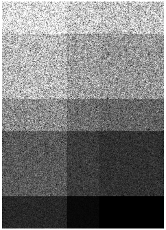
Bruit

Apparition

- erreurs générant dans les pixels des valeurs différentes des valeurs réelles
- indépendant en chaque pixel
- sources : capteur, transmission, interférences etc.
- additif, multiplicatif, impulsionnel etc.
- additif : $I(i,j) = I_0(i,j) + \eta(i,j)$



référence



B. mult. (Var=0.1)

Bruit multiplicatif

- images radar, laser
- effets photochimiques (bruit "grain")
- $I(i,j) = I_0(i,j) \cdot \eta(i,j), E[\eta] = 1$

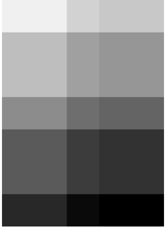
Bruit

Apparition

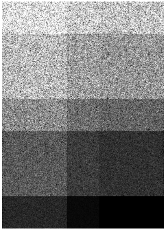
- erreurs générant dans les pixels des valeurs différentes des valeurs réelles
- indépendant en chaque pixel
- sources : capteur, transmission, interférences etc.
- additif, multiplicatif, impulsionnel etc.
- additif : $I(i,j) = I_0(i,j) + \eta(i,j)$

Bruit multiplicatif

- images radar, laser
- effets photochimiques (bruit "grain")
- $I(i,j) = I_0(i,j) \cdot \eta(i,j), E[\eta] = 1$



référence



B. mult. (Var=0.1)

Exercice :

Comment peut-on ramener le problème à un cas déjà visité ?

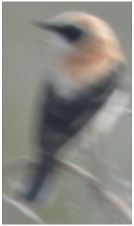
Bruit

Apparition

- erreurs générant dans les pixels des valeurs différentes des valeurs réelles
- indépendant en chaque pixel
- sources : capteur, transmission, interférences etc.
- additif, multiplicatif, impulsionnel etc.
- additif : $I(i,j) = I_0(i,j) + \eta(i,j)$

Bruit

Apparition



- erreurs générant dans les pixels des valeurs différentes des valeurs réelles
- indépendant en chaque pixel
- sources : capteur, transmission, interférences etc.
- additif, multiplicatif, impulsionnel etc.
- additif : $I(i,j) = I_0(i,j) + \eta(i,j)$

Bruit convolutif

- effet de "flou"
- défaut de mise au point
- mouvement rapide de la caméra
- $I(i,j) = I_0(i,j) \star g + \eta(i,j)$

Bruit

Apparition

- erreurs générant dans les pixels des valeurs différentes des valeurs réelles
- indépendant en chaque pixel
- sources : capteur, transmission, interférences etc.
- additif, multiplicatif, impulsionnel etc.
- additif : $I(i,j) = I_0(i,j) + \eta(i,j)$

Bruit convolutif

- effet de "flou"
- défaut de mise au point
- mouvement rapide de la caméra
- $I(i,j) = I_0(i,j) \star g + \eta(i,j)$

déconvolution
(Fergus et al.)

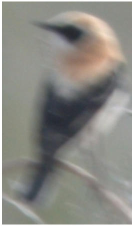
Bruit

Apparition

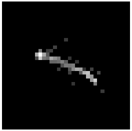
- erreurs générant dans les pixels des valeurs différentes des valeurs réelles
- indépendant en chaque pixel
- sources : capteur, transmission, interférences etc.
- additif, multiplicatif, impulsionnel etc.
- additif : $I(i,j) = I_0(i,j) + \eta(i,j)$

Bruit convolutif

- effet de "flou"
- défaut de mise au point
- mouvement rapide de la caméra
- $I(i,j) = I_0(i,j) \star g + \eta(i,j)$



déconvolution
(Fergus et al.)



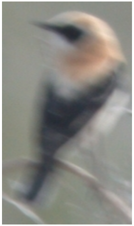
Bruit

Apparition

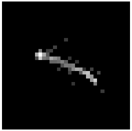
- erreurs générant dans les pixels des valeurs différentes des valeurs réelles
- indépendant en chaque pixel
- sources : capteur, transmission, interférences etc.
- additif, multiplicatif, impulsionnel etc.
- additif : $I(i,j) = I_0(i,j) + \eta(i,j)$

Bruit convolutif

- effet de "flou"
- défaut de mise au point
- mouvement rapide de la caméra
- $I(i,j) = I_0(i,j) \star g + \eta(i,j)$



déconvolution (Fergus et al.)



Filtrage

Caractéristiques

- processus qui élimine une composante indésirable d'un signal
- parfois utilisé pour créer un effet artistique etc.
- en général associé à une perte d'information
- utilise le voisinage du pixel pour calculer sa nouvelle valeur
- classifications très variées :
  - filtrage linéaire et non-linéaire
  - filtrage passe-bas, passe-bande et passe-haut
  - etc.

Filtrage linéaire

Formulation de base (1D continu)

- produit de convolution : un signal $x(t)$ et un filtre $h(t)$

$$(x \star h)(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau$$

- propriétés fondamentales : commutatif, distributif, associatif
- équivalent à un produit classique dans le domaine fréquentiel :

$$x \star h = \mathcal{F}^{-1}[\mathcal{F}(x) \cdot \mathcal{F}(h)]$$

Utilisation en TI (2D discret)

- une image I et un filtre g

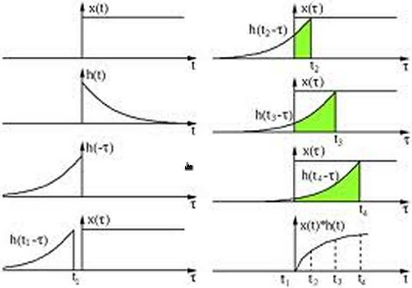
$$(I \star g)(i,j) = \sum_{n=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} I(i-n, j-m)g(n,m)$$

- en général le support de g est compact, de dimensions impaires

Filtrage linéaire

Calcul effectif en 1D continu

1. retournement : $h(\tau) \to h(-\tau)$
2. translation : $h(-\tau) \to h(t-\tau)$
3. calcul produit : $x(t) \cdot h(t-\tau)$
4. calcul intégrale : $\int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau$



Calcul effectif en TI avec exemple

- calculer le masque $w(i,j) = g(-i,-j)$ par symétrie centrale
- centrer le masque (élément $w_{0,0}$) sur le pixel courant ($I_{5,3}$)
- calcul des produits des paires correspondantes ($w_{11} \cdot I_{42}, w_{21} \cdot I_{52}$, etc.)
- faire la somme de tous les produits :

$$(I \star g)(5,3) = w_{-1,-1} \cdot I_{4,2} + \ldots + w_{1,1} \cdot I_{6,4}$$

- si le filtre conserve la moyenne de l'image $\sum w_{i,j} = 1$
- traitement particulier pour les bords

$w_{-1,-1}$	$w_{0,-1}$	$w_{1,-1}$
$w_{-1,0}$	$w_{0,0}$	$w_{1,0}$
$w_{-1,1}$	$w_{0,1}$	$w_{1,1}$

masque w 3×3

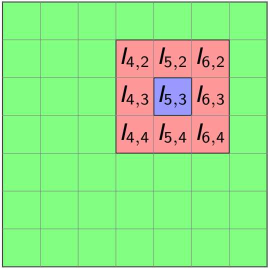


image I

$$(I \star g)(i,j) = \sum_{n=-N}^{N} \sum_{m=-M}^{M} w_{n,m} \cdot I_{i+n,j+m}$$

Filtre moyenneur

Propriétés

- le niveau de gris du pixel central est remplacé par la moyenne des niveaux de gris des pixels environnants
- filtre lisseur (donc passe-bas)
- support :

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

- image initiale et filtrage avec $l = 1$ et $l = 3$ (ou tailles 3, resp. 7 !)

Filtre moyenneur

Propriétés

- le niveau de gris du pixel central est remplacé par la moyenne des niveaux de gris des pixels environnants
- filtre lisseur (donc passe-bas)
- support :

$$w = \frac{1}{9} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

- image initiale et filtrage avec $l = 1$ et $l = 3$ (ou tailles 3, resp. 7 !)

Filtre Gaussien

Propriétés

- le niveau de gris du pixel central est remplacé par la moyenne des niveaux de gris des pixels environnants, pondérée par une Gaussienne 2D centrée dans ce pixel
- filtre lisseur (donc passe-bas)
- taille du support en fonction du paramètre σ : $l = \mathtt{Int}^{+}(3\sigma)$
- élimine moins brutalement les hautes fréquences et préserve mieux les détails
- exemple pour $\sigma = 0.625$, $l = 2$:

$$w = 0.4 \times 10^{-2} \times \begin{pmatrix} 0.03 & 0.16 & 5.98 & 0.16 & 0.03 \\ 0.16 & 7.7 & 27.8 & 7.7 & 0.16 \\ 5.98 & 27.8 & 100 & 27.8 & 5.98 \\ 0.16 & 7.7 & 27.8 & 7.7 & 0.16 \\ 0.03 & 0.16 & 5.98 & 0.16 & 0.03 \end{pmatrix}$$

Filtre Gaussien

Propriétés

- le niveau de gris du pixel central est remplacé par la moyenne des niveaux de gris des pixels environnants, pondérée par une Gaussienne 2D centrée dans ce pixel
- filtre lisseur (donc passe-bas)
- taille du support en fonction du paramètre σ : $l = \mathtt{Int}^{+}(3\sigma)$
- élimine moins brutalement les hautes fréquences et préserve mieux les détails
- exemple pour $\sigma = 0.625$, $l = 2$:

$$w = 0.4 \times 10^{-2} \times \begin{pmatrix} 0.03 & 0.16 & 5.98 & 0.16 & 0.03 \\ 0.16 & 7.7 & 27.8 & 7.7 & 0.16 \\ 5.98 & 27.8 & 100 & 27.8 & 5.98 \\ 0.16 & 7.7 & 27.8 & 7.7 & 0.16 \\ 0.03 & 0.16 & 5.98 & 0.16 & 0.03 \end{pmatrix}$$

Exercice :

Implémenter l'opération de convolution et les filtrages dans votre langage préféré

Filtre Gaussien

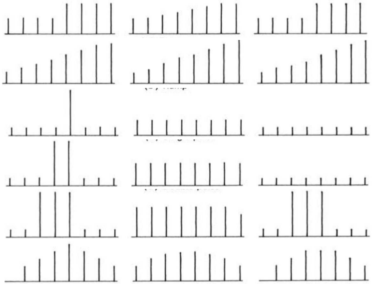
Propriétés

- le niveau de gris du pixel central est remplacé par la moyenne des niveaux de gris des pixels environnants, pondérée par une Gaussienne 2D centrée dans ce pixel
- filtre lisseur (donc passe-bas)
- taille du support en fonction du paramètre σ : $l = \mathtt{Int}^+(3\sigma)$
- élimine moins brutalement les hautes fréquences et préserve mieux les détails
- exemple pour $\sigma = 2$:

Filtre médian

Propriétés

- remplace par la valeur médiane de tous les pixels de la fenêtre d'analyse centrée sur le pixel
- filtre non-linéaire, plus coûteux
- très bien adapté au bruit impulsionnel



a) signal initial ; b) filtre moyenneur ; c) filtre médian

Exercice :

Quel est la taille du filtre médian pour avoir ces résultats ?

Filtre médian

Propriétés

- remplace par la valeur médiane de tous les pixels de la fenêtre d'analyse centrée sur le pixel
- filtre non-linéaire, plus coûteux
- très bien adapté au bruit impulsionnel


Filtre médian

Propriétés

- remplace par la valeur médiane de tous les pixels de la fenêtre d'analyse centrée sur le pixel
- filtre non-linéaire, plus coûteux
- très bien adapté au bruit impulsionnel

a) référence ; b) b. impulsionnel ; c) filtre Gaussien

Filtre médian

Propriétés

- remplace par la valeur médiane de tous les pixels de la fenêtre d'analyse centrée sur le pixel
- filtre non-linéaire, plus coûteux
- très bien adapté au bruit impulsionnel

a) référence ; b) b. impulsionnel ; c) filtre médian

Choisissez le filtrage en fonction du type du bruit et de l'application !

Filtre de Sobel

Propriétés

- les dérivées discrètes :

$$I_x[x,y] = I[x+1,y] - I[x-1,y]$$

$$I_y[x,y] = I[x,y+1] - I[x,y-1]$$

- c.a.d convolution avec $[-1\ 0\ 1]$, resp. $[-1\ 0\ 1]^T$
- sensible au bruit, lissage dans la direction orthogonale
- résultat : H_x et H_y pour les deux directions :

$$H_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \quad H_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}$$

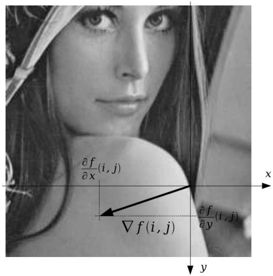
Filtre de Sobel
Propriétés

- les dérivées discrètes :

$$I_x[x, y] = I[x + 1, y] - I[x - 1, y]$$

$$I_y[x, y] = I[x, y + 1] - I[x, y - 1]$$

- c.a.d convolution avec $[-1\ 0\ 1]$, resp. $[-1\ 0\ 1]^T$
- sensible au bruit, lissage dans la direction orthogonale
- on peut calculer la magnitude du gradient : $||\nabla I|| = \sqrt{(I * H_x)^2 + (I * H_y)^2}$
- ainsi que son orientation : $\theta = \text{atan}(I * H_y)/(I * H_x)$



Filtre de Sobel

Propriétés

- les dérivées discrètes :

$$I_x[x,y] = I[x+1,y] - I[x-1,y]$$

$$I_y[x,y] = I[x,y+1] - I[x,y-1]$$

- c.a.d convolution avec $[-1\ 0\ 1]$, resp. $[-1\ 0\ 1]^T$
- sensible au bruit, lissage dans la direction orthogonale
- on peut calculer la magnitude du gradient : $||\nabla I|| = \sqrt{(I*H_x)^2 + (I*H_y)^2}$
- ainsi que son orientation : $\theta = \mathtt{atan}(I*H_y)/(I*H_x)$

Filtre de Sobel

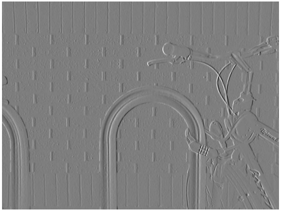
Propriétés

- les dérivées discrètes :

$$I_x[x, y] = I[x+1, y] - I[x-1, y]$$

$$I_y[x, y] = I[x, y+1] - I[x, y-1]$$

- c.a.d convolution avec $[-1\ 0\ 1]$, resp. $[-1\ 0\ 1]^T$
- sensible au bruit, lissage dans la direction orthogonale
- on peut calculer la magnitude du gradient : $||\nabla I|| = \sqrt{(I * H_x)^2 + (I * H_y)^2}$
- ainsi que son orientation : $\theta = \mathtt{atan}(I * H_y)/(I * H_x)$



Filtre de Sobel

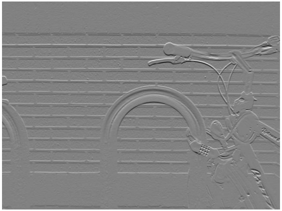
Propriétés

- les dérivées discrètes :

$$I_x[x, y] = I[x+1, y] - I[x-1, y]$$

$$I_y[x, y] = I[x, y+1] - I[x, y-1]$$

- c.a.d convolution avec $[-1\ 0\ 1]$, resp. $[-1\ 0\ 1]^T$
- sensible au bruit, lissage dans la direction orthogonale
- on peut calculer la magnitude du gradient : $||\nabla I|| = \sqrt{(I * H_x)^2 + (I * H_y)^2}$
- ainsi que son orientation : $\theta = \mathtt{atan}(I * H_y)/(I * H_x)$



Filtre de Sobel

Propriétés

- les dérivées discrètes :

$$I_x[x,y] = I[x+1,y] - I[x-1,y]$$

$$I_y[x,y] = I[x,y+1] - I[x,y-1]$$

- c.a.d convolution avec $[-1\ 0\ 1]$, resp. $[-1\ 0\ 1]^T$
- sensible au bruit, lissage dans la direction orthogonale
- on peut calculer la magnitude du gradient : $||\nabla I|| = \sqrt{(I * H_x)^2 + (I * H_y)^2}$
- ainsi que son orientation : $\theta = \mathtt{atan}(I * H_y)/(I * H_x)$

Implémentation

Court complément d'algorithmique

- nombre d'opérations élémentaires - très important
- dépend de la taille n des données
- complexité d'un algorithme : $O(f(n))$, où f est un général une combinaison de polynômes, logarithmes ou exponentielles
- signification : le nombre d'opérations effectuées est borné par $cf(n)$, lorsque n tend vers l'infini

Quelques classes de complexité :

- les algorithmes sub-linéaires, en général en $O(log(n))$. Exemple typique : recherche dichotomique
- les algorithmes en $O(n)$ et $O(nlog(n))$ sont considérés comme rapides. Exemple typique : tri optimal
- algorithmes de complexité entre $O(n^2)$ et $O(n^3)$ passent déjà moins bien a l'échelle. Exemple typique : multiplication de matrices
- au delà, on considère que les algorithmes sont impraticables

Implémentation

Séparabilité

- considérons un filtre moyenneur M de taille 7×5
- 35 opérations nécessaires par pixel

Implémentation

Séparabilité

- considérons un filtre moyenneur M de taille 7×5
- 35 opérations nécessaires par pixel
- on peut remarquer que $M = H_x \star H_y$ avec $H_x = (1/7)[1\ 1\ 1\ 1\ 1\ 1\ 1]$ et $H_y = (1/5)[1\ 1\ 1\ 1\ 1]^T$
- on peut convoluer par associativité de la manière suivante :

$$I \star M = I \star (H_x \star H_y) = (I \star H_x) \star H_y$$

- cette fois, seulement $7 + 5 = 12$ multiplications nécessaires
- dpdv. complexité, est-ce qu'on change de classe ? Pensez à un filtre de taille N $\times$ N
- méthode applicable pour Sobel, filtre gaussien etc.

Implémentation

Séparabilité

- considérons un filtre moyenneur M de taille 7×5
- 35 opérations nécessaires par pixel
- on peut remarquer que $M = H_x \star H_y$ avec $H_x = (1/7)[1\ 1\ 1\ 1\ 1\ 1\ 1]$ et $H_y = (1/5)[1\ 1\ 1\ 1\ 1]^T$
- on peut convoluer par associativité de la manière suivante :

$$I \star M = I \star (H_x \star H_y) = (I \star H_x) \star H_y$$

- cette fois, seulement $7 + 5 = 12$ multiplications nécessaires
- dpdv. complexité, est-ce qu'on change de classe ? Pensez à un filtre de taille N $\times$ N
- méthode applicable pour Sobel, filtre gaussien etc.

Exercice :

Quelle est la condition pour qu'un filtre 2D soit séparable ?

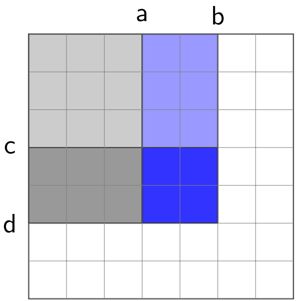
Implémentation

L'image intégrale

- une structure de données utilisée en prétraitement :

$$IN(i,j) = \sum_{n=1}^{i} \sum_{m=1}^{j} I(n,m)$$

- peut se calculer en $O(wh)$



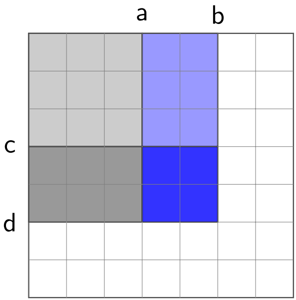
Implémentation

L'image intégrale

- une structure de données utilisée en prétraitement :

$$IN(i,j) = \sum_{n=1}^{i} \sum_{m=1}^{j} I(n,m)$$

- peut se calculer en $O(wh)$
- $\sum_{n=a}^{b} \sum_{m=c}^{d} I(n,m) = IN(b,d) - IN(a,d) - IN(b,c) + IN(a,c)$



Implémentation

L'image intégrale

- une structure de données utilisée en prétraitement :

$$IN(i,j) = \sum_{n=1}^{i} \sum_{m=1}^{j} I(n,m)$$

- peut se calculer en $O(wh)$
- $\sum_{n=a}^{b} \sum_{m=c}^{d} I(n,m) = IN(b,d) - IN(a,d) - IN(b,c) + IN(a,c)$
- 3 opérations pour calculer n'importe quelle somme de pixels
- calcul du filtrage moyenneur en $O(wh)$

Exercice :

En quelles conditions peut-on s'en servir de l'idée derrière l'image intégrale ?