

Vision par ordinateur

Séance 1 – Introduction et traitement d'images

Michaël Clément

michael.clement@enseirb-matmeca.fr

2024–25

Bordeaux INP – ENSEIRB-MATMECA

3A Intelligence Artificielle

Organisation du module

- Module de **24 heures**
 - Un peu de cours, beaucoup de pratique
 - *Domaine vaste qui évolue très vite : impossible de tout couvrir*
- Évaluation du module : **projet**
 - 6 sujets au choix, par groupes de 3 ou 4 personnes
 - Sujets ouverts : besoin d'être curieux et autonomes
 - Rendu : code et rapport au format article
 - *Présentations lors de la dernière séance*

Références

Références pour aller plus loin



Charles Deledalle, 2019 (UCSD)

Machine learning for image processing

<https://charles-deledalle.fr>



Fei-Fei Li, Justin Johnson and Serena Yeung, 2017
(Stanford)

CS231n: Convolutional Neural Networks for Visual Recognition

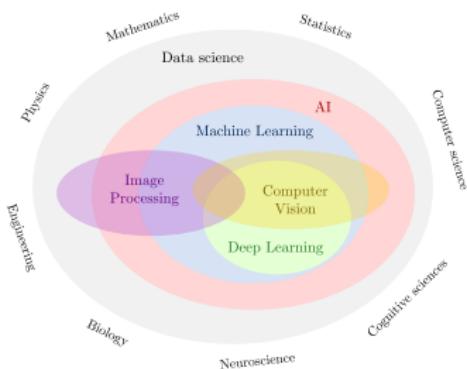
<http://cs231n.stanford.edu>

Introduction

Introduction

Vision par ordinateur ?

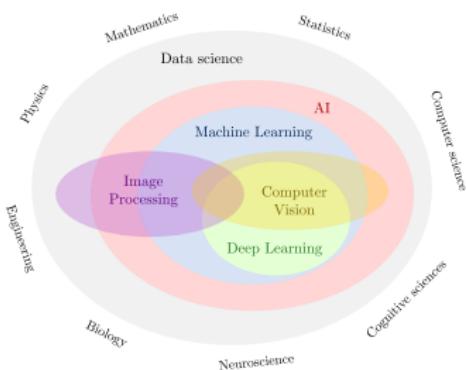
- Domaine multidisciplinaire fortement lié avec :
 - le **traitement d'images** (*image processing*)
 - l'**intelligence artificielle**



Introduction

Vision par ordinateur ?

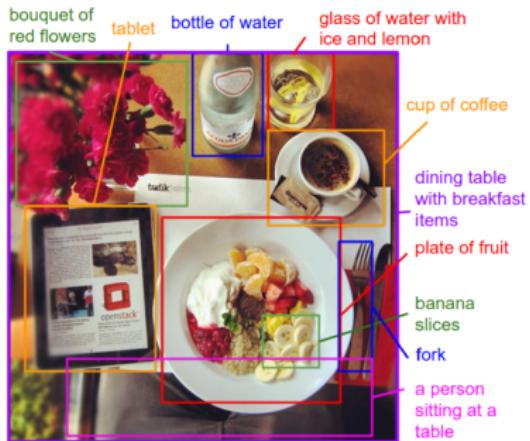
- Domaine multidisciplinaire fortement lié avec :
 - le **traitement d'images** (*image processing*)
 - l'**intelligence artificielle**



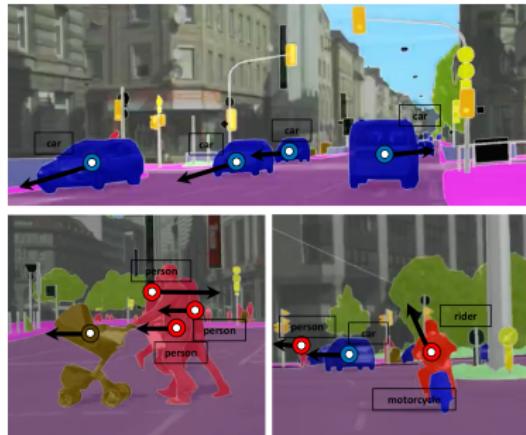
But : extraire, analyser et interpréter **automatiquement** des **informations pertinentes** des images ou des vidéos

Introduction

Quelques exemples



(Source: Luc et al., 2017)

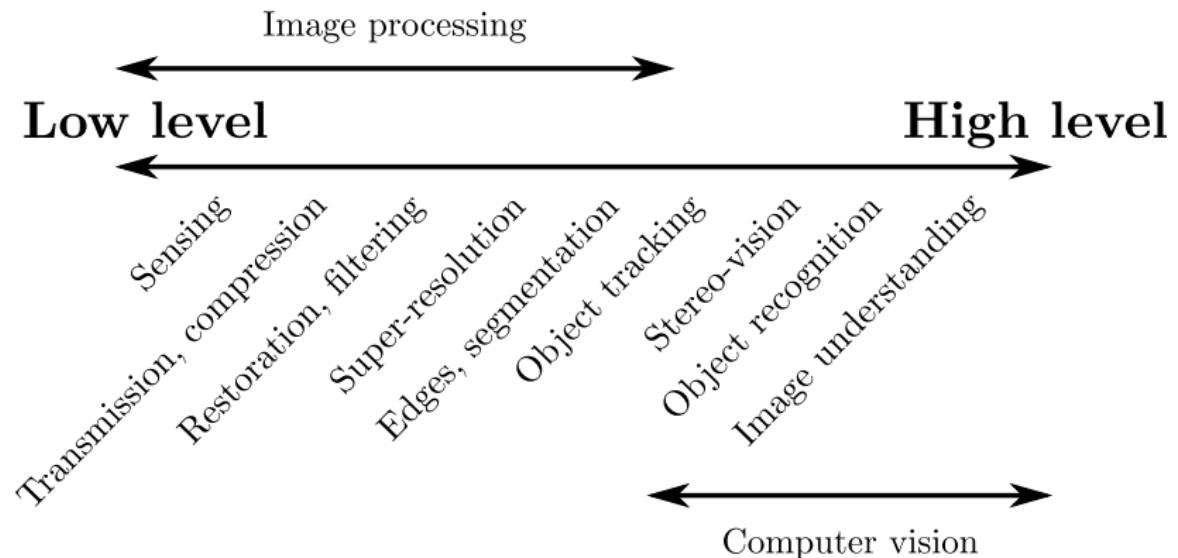


(Source: Karpathy & Fei-Fei, 2015)

Détection et reconnaissance d'objets, segmentation d'images

Traitement d'images et vision par ordinateur

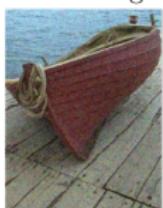
Du traitement bas niveau à l'interprétation haut niveau



Traitement d'images

Exemples de problèmes en traitement d'images

Denoising



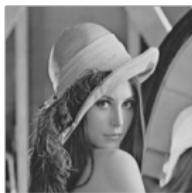
Enhancement



Compression

	ctf_2	32 KB	JPEG Image
	ctf_2	916 KB	PostScript

Feature detection



Inpainting



Super-resolution



(Source: Iasonas Kokkinos)

- Traitement d'images : nouvelle image depuis une image existante
- Traitement de vidéos : mêmes problèmes + mouvement

Traitement d'images

Exemples de problèmes en traitement d'images

Denoising



Enhancement



Compression

ctf_2	32 KB	JPEG Image
ctf_2	916 KB	PostScript

Feature detection



Inpainting



Super-resolution



(Source: Iasonas Kokkinos)

- Traitement d'images : nouvelle image depuis une image existante
- Traitement de vidéos : mêmes problèmes + mouvement

Pourquoi est-ce difficile pour l'ordinateur ?

Pourquoi est-ce difficile pour l'ordinateur ?



(a) Points de vue

Vision par ordinateur – Difficultés

Pourquoi est-ce difficile pour l'ordinateur ?



(a) Points de vue

(b) Conditions d'illumination

Vision par ordinateur – Difficultés

Pourquoi est-ce difficile pour l'ordinateur ?



(a) Points de vue

(b) Conditions d'illumination

(c) Variabilité intra-classe

Vision par ordinateur – Difficultés

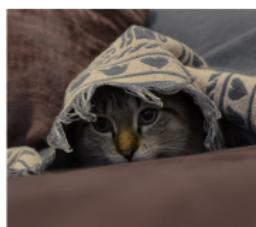
Pourquoi est-ce difficile pour l'ordinateur ?



(a) Points de vue

(b) Conditions d'illumination

(c) Variabilité intra-classe



(d) Occlusions

Vision par ordinateur – Difficultés

Pourquoi est-ce difficile pour l'ordinateur ?



(a) Points de vue

(b) Conditions d'illumination

(c) Variabilité intra-classe



(d) Occlusions

(e) Objets déformables

Fossé sémantique (*semantic gap*)



```
08 02 22 97 38 15 00 40 00 75 04 05 07 78 52 12 50 31 81  
49 49 99 40 17 81 18 57 60 87 17 40 98 43 69 48 26 56 42 00  
81 49 31 73 55 79 14 29 93 71 40 63 18 30 03 49 13 36 65  
52 70 95 23 04 60 11 42 53 68 56 01 32 54 71 37 02 36 91  
22 31 16 71 51 03 63 59 41 92 36 54 22 40 40 28 66 53 13 80  
24 47 34 20 99 03 45 02 44 75 33 53 78 36 84 20 35 17 12 50  
32 98 81 28 64 23 67 10 26 38 40 67 59 54 70 66 18 38 64 70  
67 26 20 68 02 62 12 20 95 63 94 39 63 08 40 91 66 49 94 21  
24 55 58 05 66 72 59 26 97 17 78 32 78 96 83 14 88 34 89 63 72  
21 36 23 09 75 00 76 44 20 45 35 14 00 41 33 97 34 31 33 95  
71 17 53 28 22 75 31 67 15 94 03 00 04 62 16 14 09 53 56 92  
16 39 05 42 96 35 31 47 55 58 88 24 00 17 54 24 36 29 85 57  
86 56 00 48 35 71 89 07 05 44 44 37 44 60 21 58 51 54 17 58  
19 80 81 68 05 94 47 69 28 73 92 13 84 52 17 77 04 89 55 40  
04 92 08 83 97 35 99 16 07 97 57 32 16 26 26 79 33 32 27 98 66  
44 68 87 57 62 20 72 03 66 33 67 46 55 12 32 63 93 53 69  
04 42 16 73 05 39 11 24 94 72 18 08 46 29 32 40 62 76 36  
20 49 36 41 72 30 23 88 34 05 69 82 87 59 85 74 04 36 16  
20 73 35 29 78 31 90 01 74 31 49 71 88 11 16 23 57 05 54  
01 70 54 71 83 51 58 89 16 92 33 48 81 43 32 01 59 24 40
```

What the computer sees

→
image classification
82% cat
15% dog
2% hat
1% mug

→ Fossé entre la *représentation numérique* et l'*interprétation sémantique*

Pourquoi est-ce « *facile* » pour nous ?

- Notre perception et nos raisonnements sont le fruit d'un apprentissage
 - *Mécanismes de cognition complexes*
- L'apprentissage automatique (*machine learning*) vise à reproduire ces mécanismes
 - *Apprentissage à partir d'exemples*

Pourquoi est-ce « facile » pour nous ?

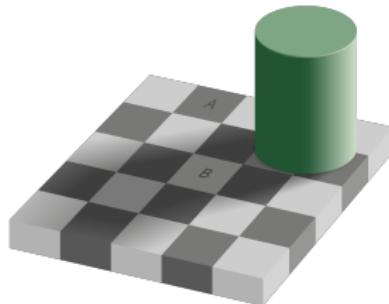
- Notre perception et nos raisonnements sont le fruit d'un apprentissage
 - *Mécanismes de cognition complexes*
- L'apprentissage automatique (*machine learning*) vise à reproduire ces mécanismes
 - *Apprentissage à partir d'exemples*

Attention, la perception humaine peut aussi être trompeuse !

Pourquoi est-ce « facile » pour nous ?

- Notre perception et nos raisonnements sont le fruit d'un apprentissage
 - *Mécanismes de cognition complexes*
- L'apprentissage automatique (*machine learning*) vise à reproduire ces mécanismes
 - *Apprentissage à partir d'exemples*

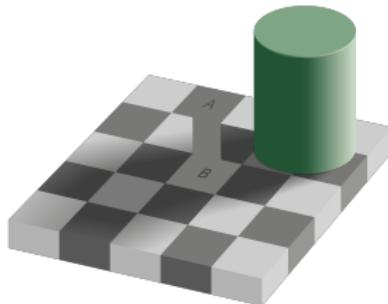
Attention, la perception humaine peut aussi être trompeuse !



Pourquoi est-ce « facile » pour nous ?

- Notre perception et nos raisonnements sont le fruit d'un apprentissage
 - *Mécanismes de cognition complexes*
- L'apprentissage automatique (*machine learning*) vise à reproduire ces mécanismes
 - *Apprentissage à partir d'exemples*

Attention, la perception humaine peut aussi être trompeuse !

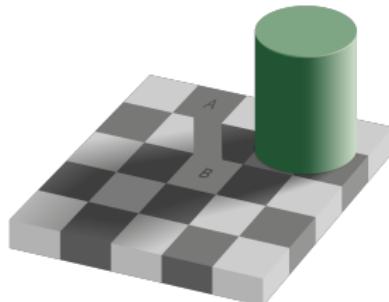


Vision par ordinateur – Difficultés

Pourquoi est-ce « facile » pour nous ?

- Notre perception et nos raisonnements sont le fruit d'un apprentissage
 - *Mécanismes de cognition complexes*
- L'apprentissage automatique (*machine learning*) vise à reproduire ces mécanismes
 - *Apprentissage à partir d'exemples*

Attention, la perception humaine peut aussi être trompeuse !

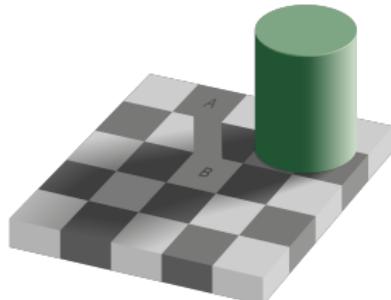


Vision par ordinateur – Difficultés

Pourquoi est-ce « facile » pour nous ?

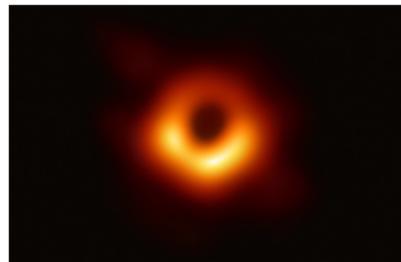
- Notre perception et nos raisonnements sont le fruit d'un apprentissage
 - *Mécanismes de cognition complexes*
- L'apprentissage automatique (*machine learning*) vise à reproduire ces mécanismes
 - *Apprentissage à partir d'exemples*

Attention, la perception humaine peut aussi être trompeuse !



Images numériques

Qu'est-ce qu'une image ?



Représentation d'une image numérique

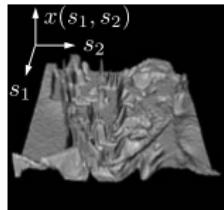
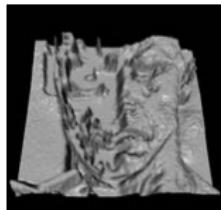
- Définition générique : fonction sur un domaine \mathcal{D} vers un espace Ω

$$I : \mathcal{D} \rightarrow \Omega$$

- En informatique, les représentations sont discrétisées et échantillonnées :

- Domaine discret $\mathcal{D} \subseteq \mathbb{Z}^n$ (nombre de points = taille de l'image)
- Valeurs échantillonnées dans $\Omega \subseteq \mathbb{R}^k$ (k = nombre de canaux)

Images 2D à niveaux de gris

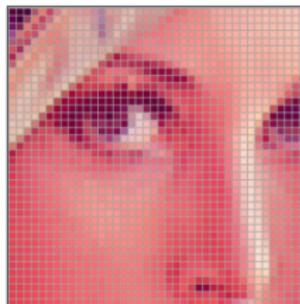
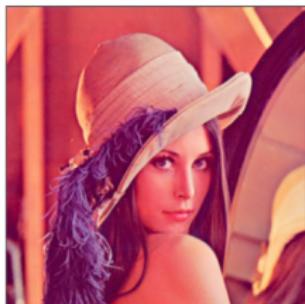


- **Deux dimensions** : domaine $\mathcal{D} \subseteq \mathbb{Z}^2$
 - Par exemple $\mathcal{D} = \{0, \dots, 27\} \times \{0, \dots, 27\}$
 - Image de taille 28×28
- **Niveaux de gris** : valeurs dans $\Omega \subseteq \mathbb{N}$
 - Usuellement $\Omega = \{0, \dots, 255\}$ (un octet, 256 valeurs possibles)
 - Convention : du plus sombre (0) au plus lumineux (255)

Sur machine : tableau (matrice) de nombres entiers
→ Chaque élément est appelé ***pixel***

Images 2D

Images 2D en couleurs

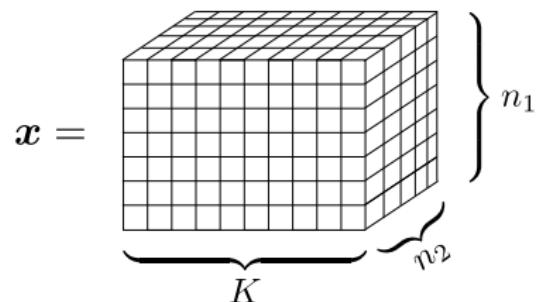


139	162	119	98	127	202
88	88	44	27	37	160
95	121	83	71	106	184
133	124	110	105	159	218
54	42	52	38	107	185
66	80	74	62	143	204
127	107	116	145	200	226
47	26	40	89	160	198
86	69	86	128	187	210
112	122	137	186	220	229
39	53	79	145	189	199
82	98	120	175	207	207
128	162	186	208	220	222
89	107	144	179	194	190
107	149	180	201	207	195
169	192	206	220	219	224
117	148	170	189	187	187
156	171	182	195	192	194

- Toujours **2D** : domaine $\mathcal{D} \subseteq \mathbb{Z}^2$
- **En couleurs** : valeurs dans $\Omega \subseteq \mathbb{N}^3$
 - Trois canaux : *Red*, *Green*, et *Blue* (*RGB*)
 - Chaque *pixel* est représenté par 3 octets
 - Il existe d'autres espaces de couleur : *HSV*, *Lab*, *YUV*, etc.

Images multidimensionnelles

- Une image de taille $n_1 \times n_2$ avec k canaux peut être représentée par un **tableau multidimensionnel** :



- En NumPy : `np.ndarray(shape=(n1, n2, k))`
- En *deep learning* : **tenseurs**

Images spectrales

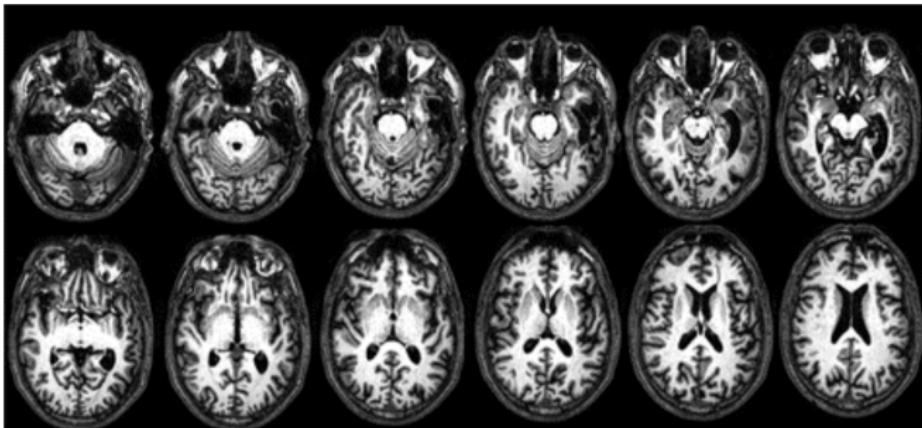
Images spectrales



- Chacun des k canaux représente une bande de fréquences
- Pour $k \approx 10$: imagerie multispectrale
- Pour $k \approx 200$: imagerie hyperspectrale
- Utilisées par exemple en astronomie, télédétection, surveillance, etc.

Images volumiques

Volumes 3D



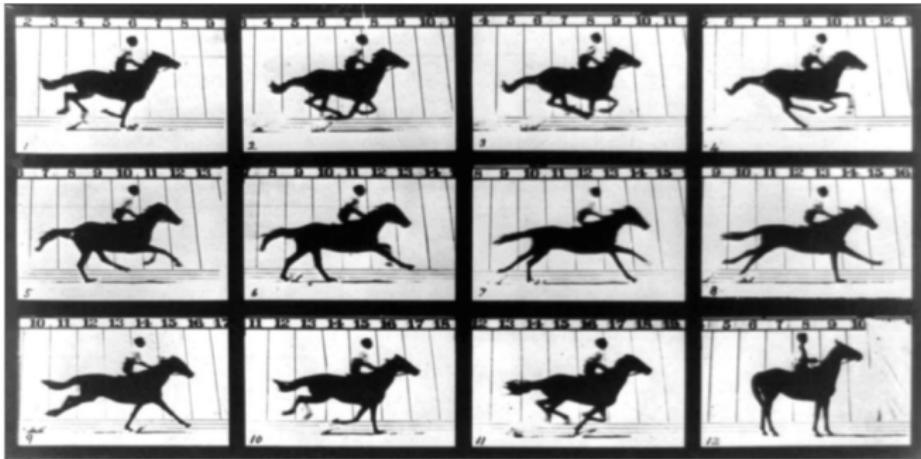
Coupes IRM à différentes profondeurs

Scan 3D du cerveau

- Espace 3D : domaine $\mathcal{D} \subseteq \mathbb{Z}^3$
- Les pixels 3D sont appelés ***voxels***

Vidéos

Vidéos : 2D+t



The Horse in Motion (1878, Eadweard Muybridge)

Vidéos : séquences d'images

- 2 dimensions pour l'espace
- 1 dimension pour le temps

Quels traitements appliquer sur les images ?

Deux grandes familles de transformations

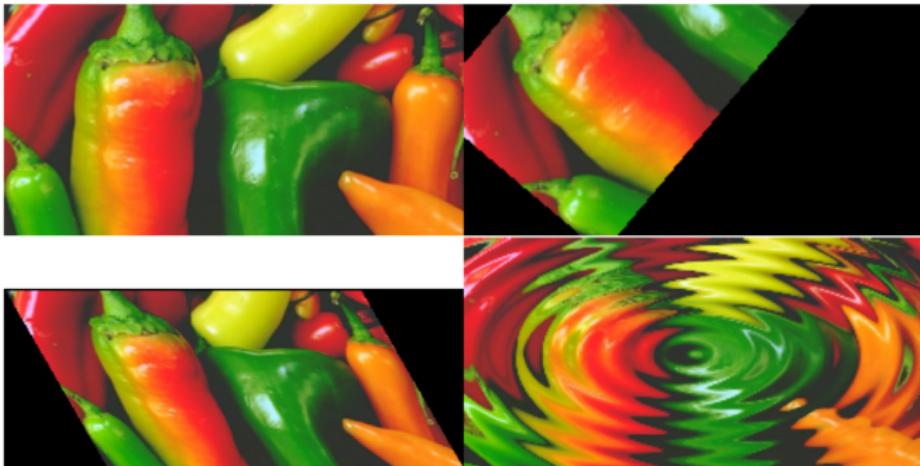
- **Géométriques** : modifier la position des pixels
- **Colorimétriques** : modifier la valeur des pixels

Différents types de transformations

- **Ponctuelles** : pixel par pixel
- **Locales** : en fonction d'un voisinage de pixels
- **Globales** : transformation de toute l'image

Transformations géométriques

Modifier la position des pixels

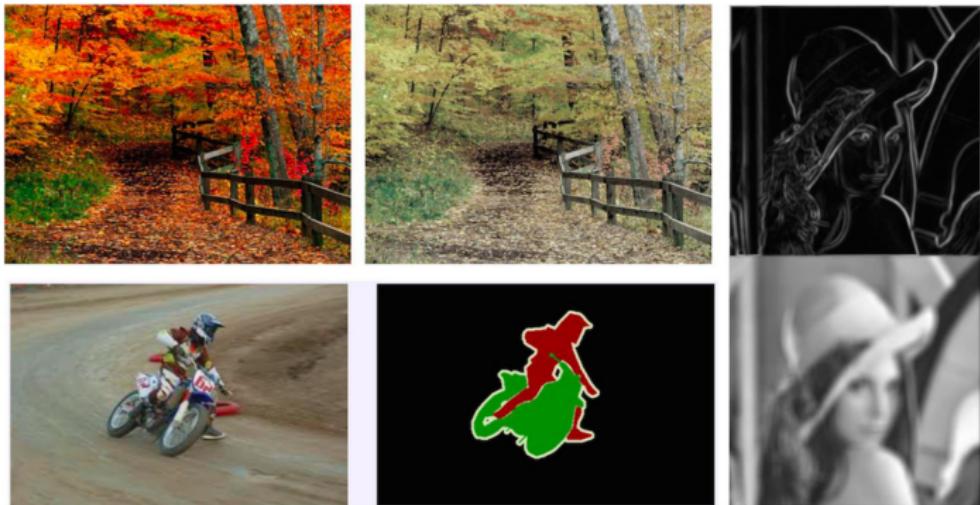


Quelques exemples

- Recalage d'images, création de panoramas
- Calibration de caméra, stéréoscopie, caméras *fisheye*

Transformations colorimétriques

Modifier la valeur des pixels



Quelques exemples

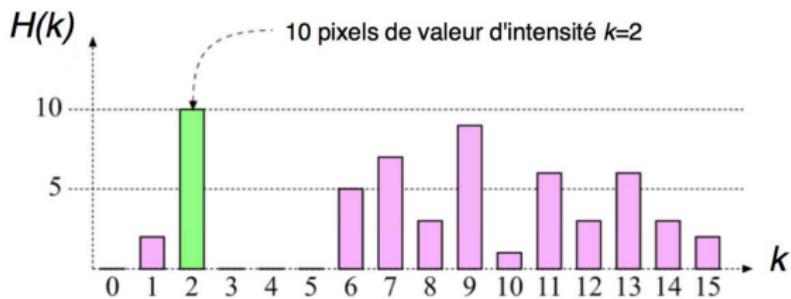
- Changement de contraste, filtrage (flou, débruitage, etc.)
- *Au sens large* : détection de contours, seuillage, segmentation

Transformations ponctuelles : histogramme et seuillage

Histogramme d'une image

Histogramme d'une image

- Description de la répartition des niveaux de gris
- $H(k) = \#\{(i, j) \in M \times N : I(i, j) = k\}$
- Extensible à la couleur : un histogramme pour chaque canal



$H(k)$	0	2	10	0	0	0	5	7	3	9	1	6	3	6	3	2
k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

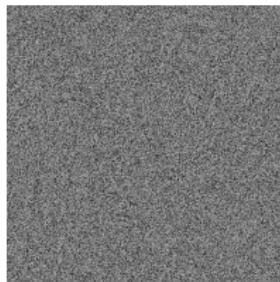
Histogramme d'une image

L'histogramme ne contient aucune information spatiale

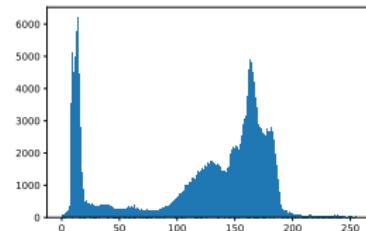
- Deux images différentes peuvent avoir le même histogramme
- Impossible de reconstruire une image à partir de son histogramme



(a) Image



(b) Pixels mélangés



(c) Même histogramme

Histogramme d'une image

Information de contraste

- L'histogramme d'une image nous informe sur son **contraste** :
 - *Image sombre* : valeurs proches de 0
 - *Image claire* : valeurs proches de 255
 - *Contraste faible* : valeurs tassées
 - *Contraste élevé* : valeur réparties
- L'histogramme peut être calculé localement (zone d'intérêt)

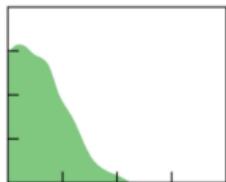


image sombre

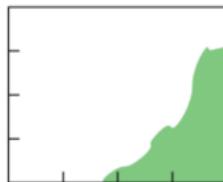


image claire

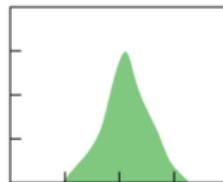


image peu contrastée

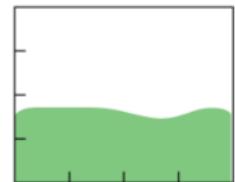


image très contrastée

Exemple : étirement de l'histogramme

Étirement d'histogramme

- Principe : répartir les fréquences d'apparition des pixels sur toute la largeur de l'histogramme
- Normalisation des valeurs des pixels dans l'intervalle $[0, 255]$:

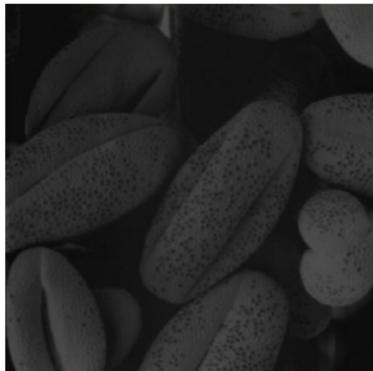
$$I'(i, j) = 0 + 255 \times \frac{I(i, j) - \min I}{\max I - \min I}$$

Exemple : étirement de l'histogramme

Étirement d'histogramme

- Principe : répartir les fréquences d'apparition des pixels sur toute la largeur de l'histogramme
- Normalisation des valeurs des pixels dans l'intervalle $[0, 255]$:

$$I'(i, j) = 0 + 255 \times \frac{I(i, j) - \min I}{\max I - \min I}$$

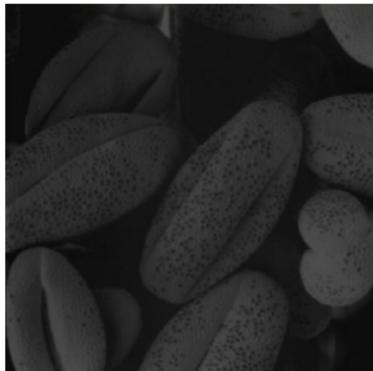


Exemple : étirement de l'histogramme

Étirement d'histogramme

- Principe : répartir les fréquences d'apparition des pixels sur toute la largeur de l'histogramme
- Normalisation des valeurs des pixels dans l'intervalle $[0, 255]$:

$$I'(i, j) = 0 + 255 \times \frac{I(i, j) - \min I}{\max I - \min I}$$



Exemple : égalisation de l'histogramme

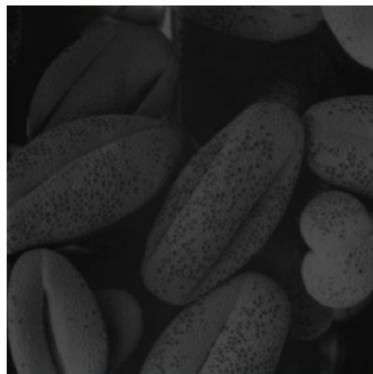
Égalisation d'histogramme

- Ajustement du contraste par uniformisation des valeurs des pixels
- Calcul à partir de l'histogramme cumulé $C(k) = \sum_{i=0}^k H(i)$ où $H(k)$ est l'histogramme normalisé
- Les pixels de valeur k auront pour nouvelle valeur $255 \times C(k)$

Exemple : égalisation de l'histogramme

Égalisation d'histogramme

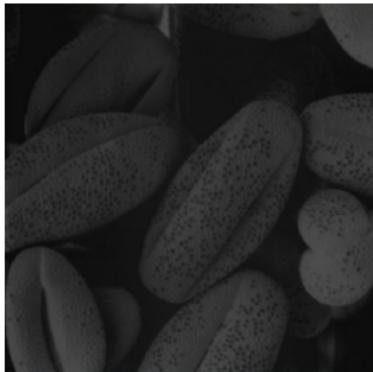
- Ajustement du contraste par uniformisation des valeurs des pixels
- Calcul à partir de l'histogramme cumulé $C(k) = \sum_{i=0}^k H(i)$ où $H(k)$ est l'histogramme normalisé
- Les pixels de valeur k auront pour nouvelle valeur $255 \times C(k)$



Exemple : égalisation de l'histogramme

Égalisation d'histogramme

- Ajustement du contraste par uniformisation des valeurs des pixels
- Calcul à partir de l'histogramme cumulé $C(k) = \sum_{i=0}^k H(i)$ où $H(k)$ est l'histogramme normalisé
- Les pixels de valeur k auront pour nouvelle valeur $255 \times C(k)$



Seuillage d'une image

Seuillage d'une image

- Fixer la valeur de certains pixels en fonction d'un **seuil** (*threshold*)
- Exemple : **binarisation** selon un seuil T (séparation *fond/forme*)

$$I_T(i, j) = \begin{cases} 255 & \text{si } I(i, j) \leq T, \\ 0 & \text{sinon.} \end{cases}$$

Seuillage d'une image

Seuillage d'une image

- Fixer la valeur de certains pixels en fonction d'un **seuil** (*threshold*)
- Exemple : **binarisation** selon un seuil T (séparation *fond/forme*)

$$I_T(i, j) = \begin{cases} 255 & \text{si } I(i, j) \leq T, \\ 0 & \text{sinon.} \end{cases}$$

Méthode d'**Otsu** : détermination automatique d'un seuil optimal
→ Valeur T qui minimise la variance intra-classe

Seuillage d'une image

Seuillage d'une image

- Fixer la valeur de certains pixels en fonction d'un **seuil** (*threshold*)
- Exemple : **binarisation** selon un seuil T (séparation *fond/forme*)

$$I_T(i, j) = \begin{cases} 255 & \text{si } I(i, j) \leq T, \\ 0 & \text{sinon.} \end{cases}$$

Méthode d'**Otsu** : détermination automatique d'un seuil optimal
→ Valeur T qui minimise la variance intra-classe



$T = 119$



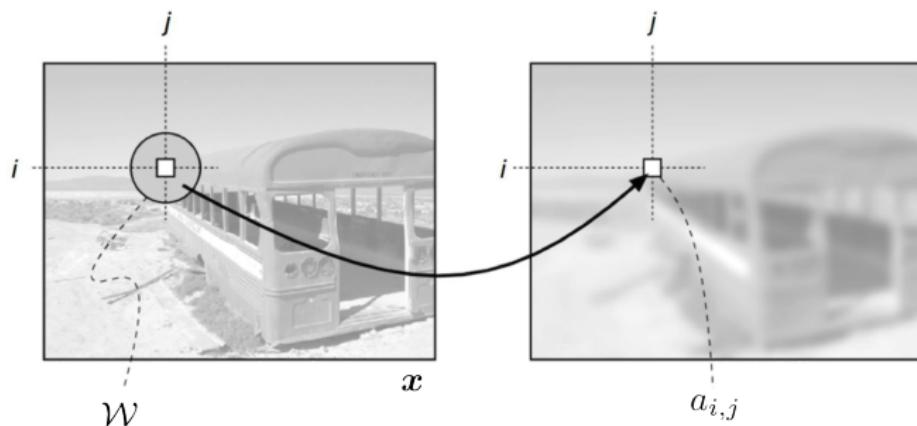
שיטה זו מושגת באמצעות נזירובץ (thresholding).
זהו ערך מסוים ממבר ליעמיך
פמייך אל תיש איש את אויה בכם
ב אחר הרובל תקינה מאת עמיית
כאותם מכבר לך לפ' רבע השניות
וז ולפי מעת השיים תמעיט מכך
ר' תבוארה מאכלהך יא עיריה א
ע' ויראות מאכלהך יא עני יהוה א
ל' אמרת מטה פטני נאת משפטני' העשורה
ה' הארכ' פראה ישכחים על הארכ' כ-
ה' הארכ' פראה ואכלדם לשבעי' זי

$T = 101$

Transformations locales : filtrage spatial

Filtrage spatial

Filtrage spatial d'une image



Transformation locale

- Principe : combiner la valeur du pixel $I(i, j)$ avec un voisinage local défini autour de (i, j)

Convolution 2D

Filtrage linéaire : convolution 2D

- Convolution $I * \kappa$ d'une image I par un filtre κ

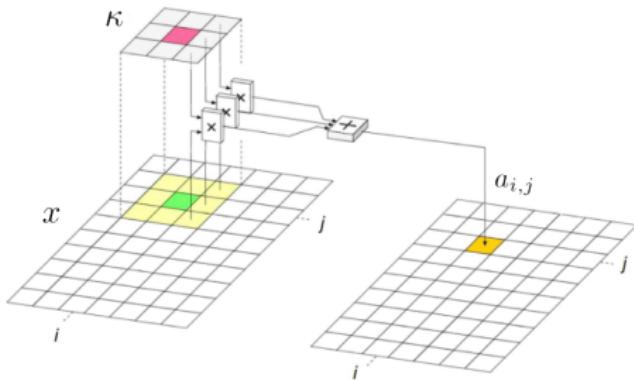
$$(I * \kappa)(i, j) = \sum_{u=-\infty}^{+\infty} \sum_{v=-\infty}^{+\infty} I(i+u, j+v) \kappa(u, v)$$

Convolution 2D

Filtrage linéaire : convolution 2D

- Convolution $I * \kappa$ d'une image I par un filtre κ

$$(I * \kappa)(i, j) = \sum_{u=-\infty}^{+\infty} \sum_{v=-\infty}^{+\infty} I(i+u, j+v) \kappa(u, v)$$

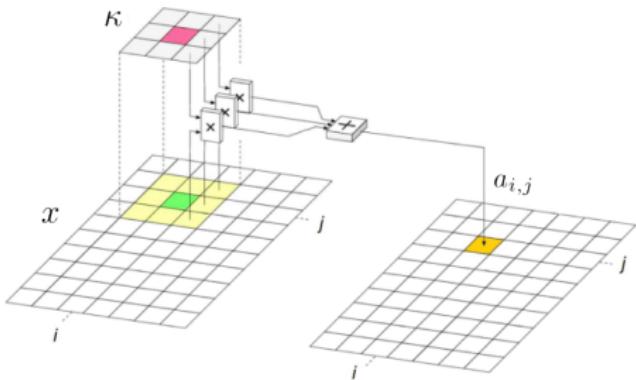


Convolution 2D

Filtrage linéaire : convolution 2D

- Convolution $I * \kappa$ d'une image I par un filtre κ

$$(I * \kappa)(i, j) = \sum_{u=-\infty}^{+\infty} \sum_{v=-\infty}^{+\infty} I(i+u, j+v) \kappa(u, v)$$



Ici κ est de taille $S = 3 \times 3$

$$\Rightarrow \sum_{u=-\infty}^{+\infty} \sum_{v=-\infty}^{+\infty} \equiv \sum_{u=-1}^{+1} \sum_{v=-1}^{+1}$$

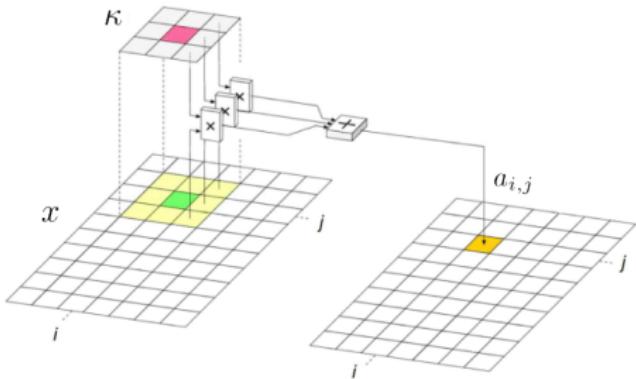
κ est centré en (i, j)

Convolution 2D

Filtrage linéaire : convolution 2D

- Convolution $I * \kappa$ d'une image I par un filtre κ

$$(I * \kappa)(i, j) = \sum_{u=-\infty}^{+\infty} \sum_{v=-\infty}^{+\infty} I(i+u, j+v) \kappa(u, v)$$



Ici κ est de taille $S = 3 \times 3$

$$\Rightarrow \sum_{u=-\infty}^{+\infty} \sum_{v=-\infty}^{+\infty} \equiv \sum_{u=-1}^{+1} \sum_{v=-1}^{+1}$$

κ est centré en (i, j)

Cette formulation correspond en fait à une corrélation croisée \star (propriétés mathématiques différentes).
Pour une « vraie » convolution : $I(i-u, j-v)$ et noyau retourné $k(-u, -v)$.

Filtre moyenneur

Exemple : filtre moyenneur (*box filter*)

- Moyenne des pixels dans le voisinage, par exemple pour $S = 3 \times 3$:

$$\kappa = \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & \mathbf{1/9} & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix} = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & \mathbf{1} & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Filtre moyenneur

Exemple : filtre moyenneur (*box filter*)

- Moyenne des pixels dans le voisinage, par exemple pour $S = 3 \times 3$:

$$\kappa = \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & \mathbf{1/9} & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix} = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & \mathbf{1} & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

→ *Applique un flou sur l'image*

Filtre moyenneur

Exemple : filtre moyenneur (*box filter*)

- Moyenne des pixels dans le voisinage, par exemple pour $S = 3 \times 3$:

$$\kappa = \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & \mathbf{1/9} & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix} = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & \mathbf{1} & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

→ *Applique un flou sur l'image*



Image initiale

Filtre moyenneur

Exemple : filtre moyenneur (*box filter*)

- Moyenne des pixels dans le voisinage, par exemple pour $S = 3 \times 3$:

$$\kappa = \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & \mathbf{1/9} & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix} = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & \mathbf{1} & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

→ *Applique un flou sur l'image*



Image initiale

Filtre moyenneur 3×3

Filtre moyenneur

Exemple : filtre moyenneur (*box filter*)

- Moyenne des pixels dans le voisinage, par exemple pour $S = 3 \times 3$:

$$\kappa = \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & \mathbf{1/9} & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix} = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & \mathbf{1} & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

→ *Applique un flou sur l'image*



Image initiale

Filtre moyenneur 5×5

Filtre moyenneur

Exemple : filtre moyenneur (*box filter*)

- Moyenne des pixels dans le voisinage, par exemple pour $S = 3 \times 3$:

$$\kappa = \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & \mathbf{1/9} & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix} = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & \mathbf{1} & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

→ *Applique un flou sur l'image*



Image initiale

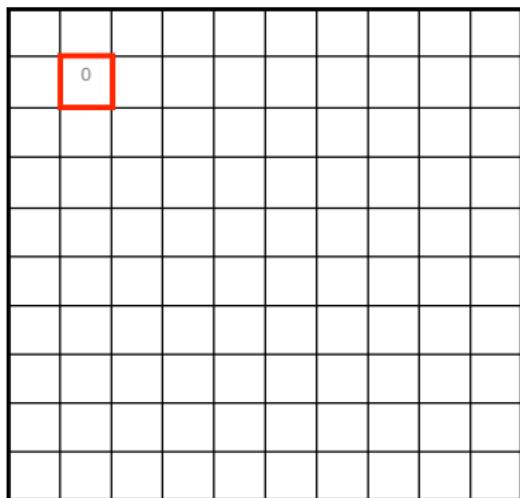
Filtre moyenneur 7×7

Filtre moyenneur

Déroulement du calcul

$$\text{Filtre } \kappa = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & \underline{1} & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

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



Filtre moyenneur

Déroulement du calcul

$$\text{Filtre } \kappa = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & \underline{1} & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

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

0											

Filtre moyenneur

Déroulement du calcul

$$\text{Filtre } \kappa = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & \underline{1} & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

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

0	10	20							

Filtre moyenneur

Déroulement du calcul

$$\text{Filtre } \kappa = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & \underline{1} & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

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

			0	10	20	30			

Filtre moyenneur

Déroulement du calcul

$$\text{Filtre } \kappa = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & \underline{1} & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

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

Filtre moyenneur

Déroulement du calcul

$$\text{Filtre } \kappa = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & \underline{1} & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

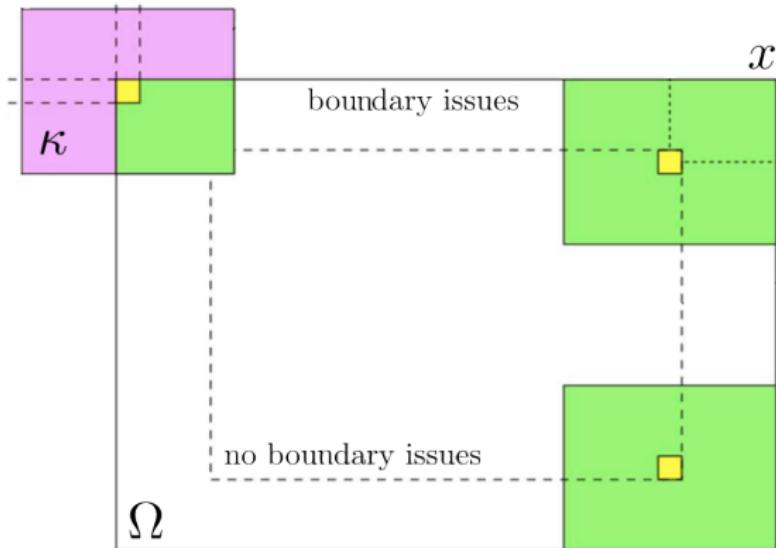
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	0	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

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

Gestion des bords de l'image

Gestion des bords de l'image

Comment gérer les cas où le filtre κ sort du domaine de l'image ?



Gestion des bords de l'image

Quelques stratégies

Agrandir l'image selon différentes stratégies avant d'appliquer le filtre :



zero-padding



extension



mirror



periodical

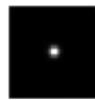
Filtre gaussien

Exemple : filtre gaussien

$$\text{Filtre gaussien: } \kappa(i, j) = \frac{1}{Z} \exp\left(-\frac{i^2 + j^2}{2\sigma^2}\right)$$



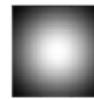
...



σ petit



σ moyen



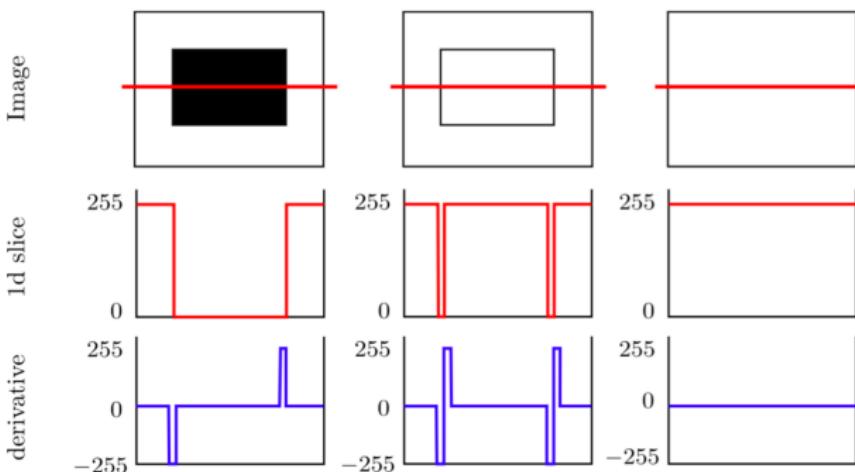
σ grand

- $i^2 + j^2$: distance au pixel central ;
- σ : largeur de la gaussienne (quantité de flou) ;
- Z : normalisation telle que $\sum \kappa_{i,j} = 1$.

Détection de contours

Qu'est-ce qu'un contour ?

- **Contours** : zones de l'image qui permettent de délimiter les objets
- Caractérisés par des **changements d'intensité**
 - *Dérivées partielles* de l'image (fonction 2D)



Gradient ∇I d'une image

- Gradient de l'image défini par le vecteur $\nabla I = [\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}]^T$
- L'image est discrète, le gradient peut être approximé par différences finies, par exemple : $\frac{\partial I}{\partial x}(i, j) \approx \frac{1}{2}(I(i + 1, j) - I(i - 1, j))$

Gradient ∇I d'une image

- Gradient de l'image défini par le vecteur $\nabla I = [\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}]^T$
- L'image est discrète, le gradient peut être approximé par différences finies, par exemple : $\frac{\partial I}{\partial x}(i, j) \approx \frac{1}{2}(I(i + 1, j) - I(i - 1, j))$

Finalement, cela revient à effectuer des **convolutions 2D** :

Détection de contours

Gradient ∇I d'une image

- Gradient de l'image défini par le vecteur $\nabla I = [\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}]^T$
- L'image est discrète, le gradient peut être approximé par différences finies, par exemple : $\frac{\partial I}{\partial x}(i, j) \approx \frac{1}{2}(I(i + 1, j) - I(i - 1, j))$

Finalement, cela revient à effectuer des **convolutions 2D** :

Gradient **horizontal** $I * \kappa_h$ avec

$$\kappa_h = \frac{1}{2} \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

Détection de contours

Gradient ∇I d'une image

- Gradient de l'image défini par le vecteur $\nabla I = [\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}]^T$
- L'image est discrète, le gradient peut être approximé par différences finies, par exemple : $\frac{\partial I}{\partial x}(i, j) \approx \frac{1}{2}(I(i + 1, j) - I(i - 1, j))$

Finalement, cela revient à effectuer des **convolutions 2D** :

Gradient **horizontal** $I * \kappa_h$ avec

$$\kappa_h = \frac{1}{2} \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$



Détection de contours

Gradient ∇I d'une image

- Gradient de l'image défini par le vecteur $\nabla I = [\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}]^T$
- L'image est discrète, le gradient peut être approximé par différences finies, par exemple : $\frac{\partial I}{\partial x}(i, j) \approx \frac{1}{2}(I(i + 1, j) - I(i - 1, j))$

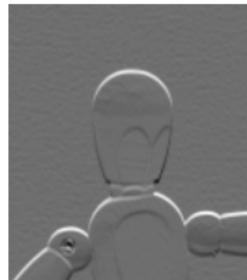
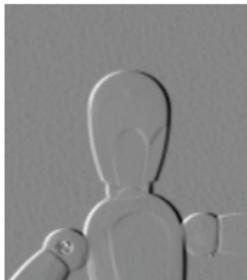
Finalement, cela revient à effectuer des **convolutions 2D** :

Gradient **horizontal** $I * \kappa_h$ avec

$$\kappa_h = \frac{1}{2} \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

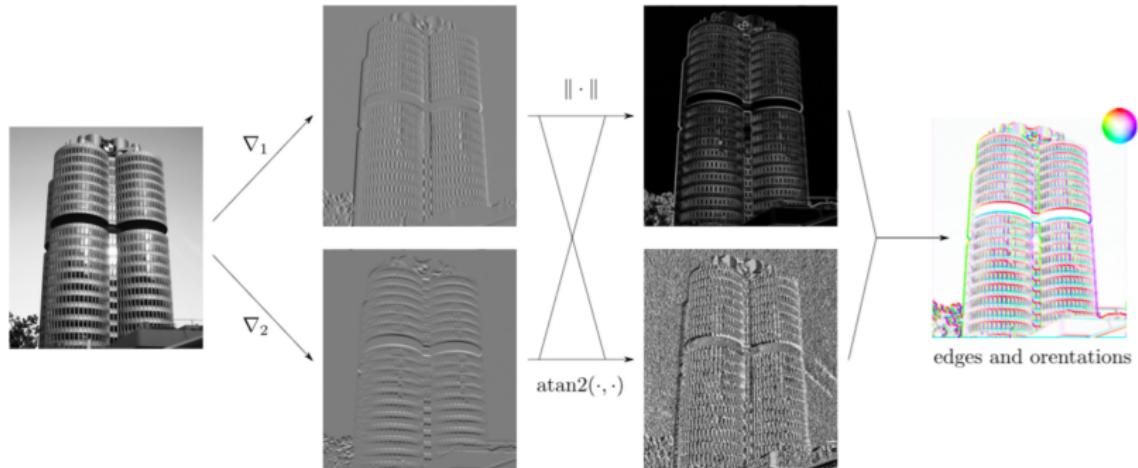
Gradient **vertical** $I * \kappa_v$ avec

$$\kappa_v = \frac{1}{2} \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$



Détection de contours

Intensité et orientation



- **Intensité** des contours : norme $\|\nabla I\| = \sqrt{\nabla_1 I^2 + \nabla_2 I^2}$
- **Orientation** des contours : $\text{atan2}(\nabla_2 I, \nabla_1 I)$

→ Détection : seuillage de l'image des intensités

Détection de contours

Filtres pour la détection de contours

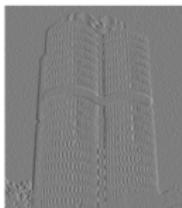
Filtre de Prewitt

$$\kappa_h = \frac{1}{2} \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \text{ et}$$

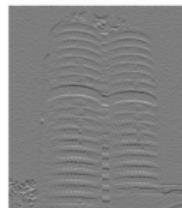
$$\kappa_v = \frac{1}{2} \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$



(a) x



(b) $\nabla_1 x$



(c) $\nabla_2 x$



(d) $\|\nabla x\|$

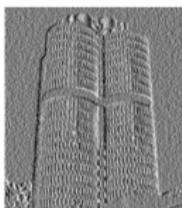
Filtre de Sobel

$$\kappa_h = \frac{1}{2} \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \text{ et}$$

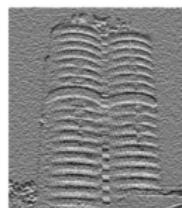
$$\kappa_v = \frac{1}{2} \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$



(e) x



(f) $\nabla_1 x$ (Prewitt)



(g) $\nabla_2 x$ (Prewitt)



(h) $\|\nabla x\|$ (Sobel)

→ Moins sensibles au bruit

Conclusion

Pour conclure...

- **Panorama sur quelques techniques de traitement d'images :**
 - Différent types d'images numériques
 - Manipulation de l'histogramme, seuillage
 - Filtrage spatial par convolution
 - Détection de contours

Pour conclure...

- **Panorama sur quelques techniques de traitement d'images :**
 - Différent types d'images numériques
 - Manipulation de l'histogramme, seuillage
 - Filtrage spatial par convolution
 - Détection de contours
- **Nombreux sujets non abordés :**
 - Passage du 3D au 2D (modèles de caméra, vision stéréoscopique)
 - Filtrage fréquentiel (transformée de Fourier)
 - Extraction de caractéristiques (points d'intérêt SIFT, SURF, etc.)
 - Et plein d'autres choses...

Pour conclure...

- **Panorama sur quelques techniques de traitement d'images :**
 - Différent types d'images numériques
 - Manipulation de l'histogramme, seuillage
 - Filtrage spatial par convolution
 - Détection de contours
- **Nombreux sujets non abordés :**
 - Passage du 3D au 2D (modèles de caméra, vision stéréoscopique)
 - Filtrage fréquentiel (transformée de Fourier)
 - Extraction de caractéristiques (points d'intérêt SIFT, SURF, etc.)
 - Et plein d'autres choses...

→ *La prochaine fois* : approches par **deep learning**

Questions ?

Acknowledgments:

C. Deledalle

V. Lepetit

V.-T. Ta