

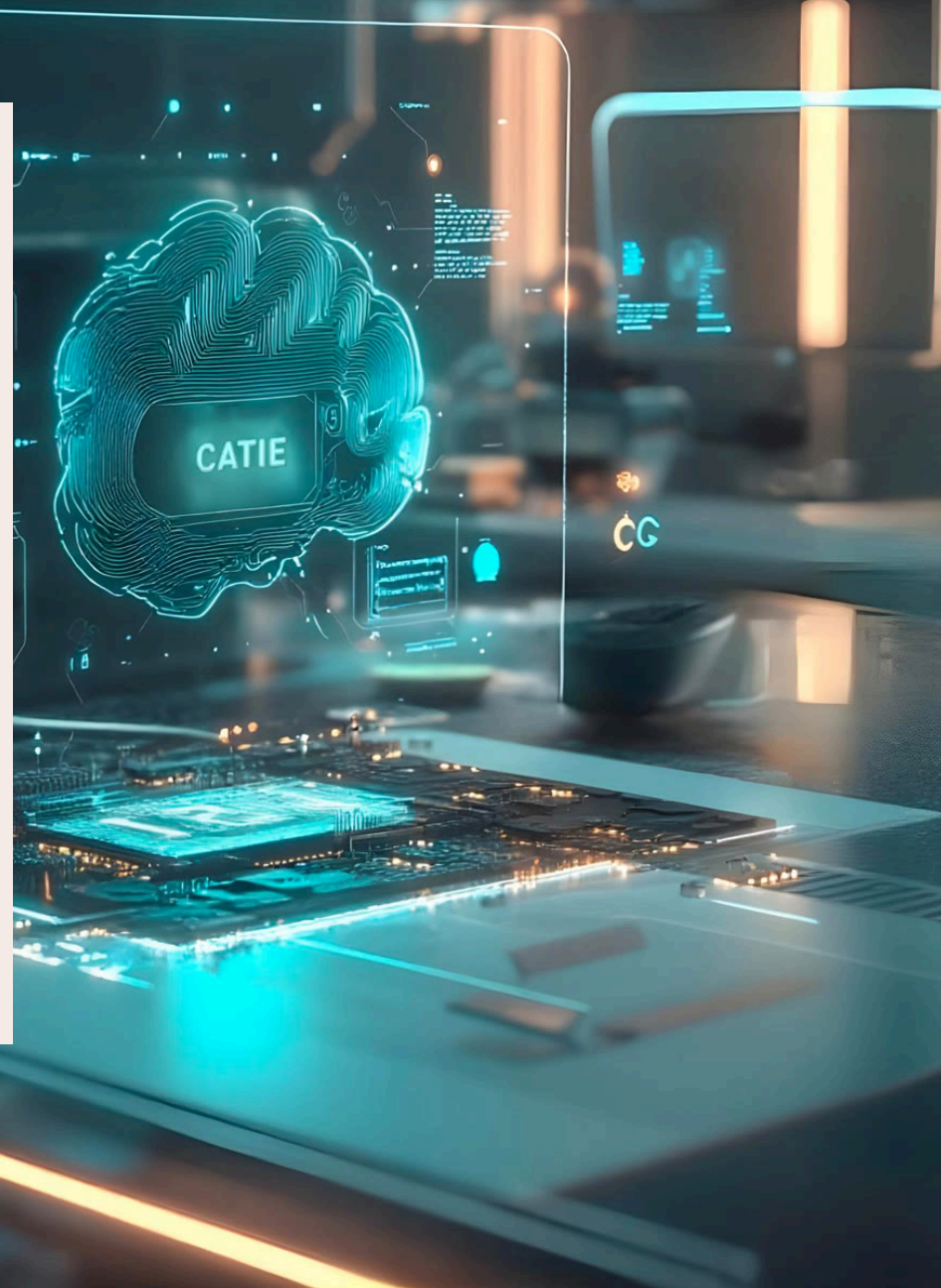


# CATIE

## Traitement d'images

Traitement : outils et méthodes

2026





# 1. Principes et manipulation

À quoi sert de traiter une image ?

# Qu'est-ce que le traitement d'image ?

- Manipulation d'images numériques
- Modifier / corriger l'apparence d'une image
- Mise en évidence de caractéristiques (éliminer ce qui est inutile, accentuer les informations utiles)

**Objectif : se mettre dans des conditions idéales et répétables pour l'analyse de l'image**

$$f(\text{image}) = \text{image}$$

# Principales problématiques du traitement d'image

- **Changement d'espace de couleur** (Ex: RGB2GRAY, RGB2HSV, ...)
- **Redimensionnement** : ajustement de la taille de l'image
- **Flou** : améliorer la netteté de l'image
- **Bruit** : supprimer les artefacts indésirables
- **Contraste** : améliorer la visibilité des détails
- **Luminosité** : améliorer la visibilité des objets
- **Détection de contours** : mettre en évidence les contours des objets
- **Détection de mouvements**
- ...

# Types d'images (entrée et sortie)

- **Couleur** : combinaison de 3 canaux
  - Utilisé pour la reconnaissance d'objets, la détection de mouvement, ...
  - Traitements possibles : cf niveaux de gris
- **Niveaux de gris** : intensité lumineuse 0-255 (ou profondeur)
  - Utilisé pour la détection de contours, la classification, la reconnaissance de formes, détection de mouvements, ...
  - Traitements possibles : segmentation, égalisation, seuillage, filtrage
- **Binaire** : noir et blanc (0 ou 1)
  - Utilisé pour la segmentation (masque, seuillage)
  - Traitements possibles : érosion, dilatation, ouverture, fermeture

# Outils les plus courants

- **OpenCV** : bibliothèque open-source de vision par ordinateur
- **Scikit-image** : bibliothèque Python (plus orientée traitement d'image)
- **Pillow** : idem

# Manipulation d'images

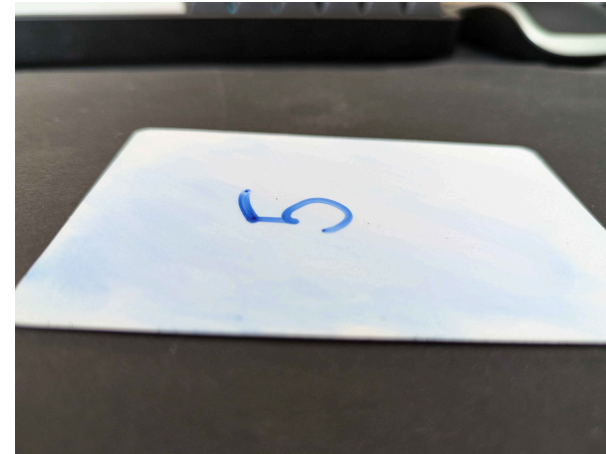
- Convertir une image dans un autre espace de couleur
  - Redimensionner, rogner, pivoter une image
  - Warping : déformation d'une image (ex: correction de perspective)
- Attention à la perte de qualité lors de la manipulation d'images

# Exercice n°1

- Nous voulons utiliser un modèle d'OCR entraîné sur une BDD MNIST
- Les images sont en niveaux de gris (28x28x1), chiffre blanc sur fond noir, centrées, le chiffre prend un maximum de place
- Vous devez **traiter** vos images pour les rendre compatibles avec le modèle

## À faire :

- Convertir l'image en niveaux de gris
- Corriger la perspective
- Rogner l'image autour du chiffre



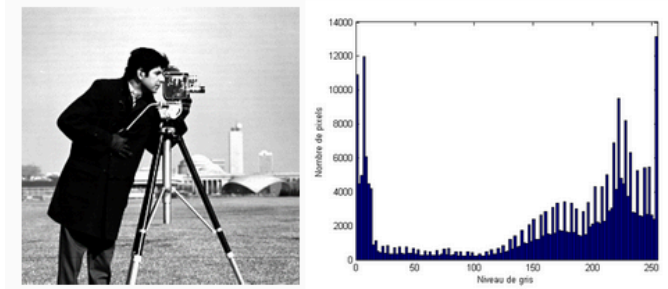
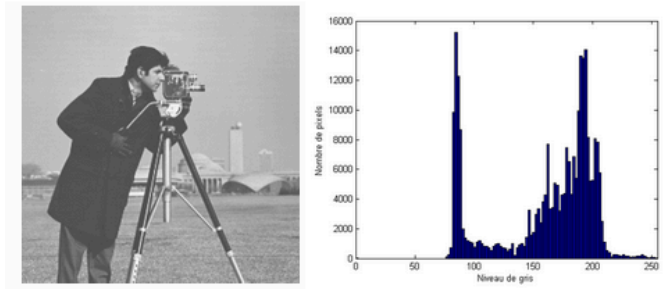
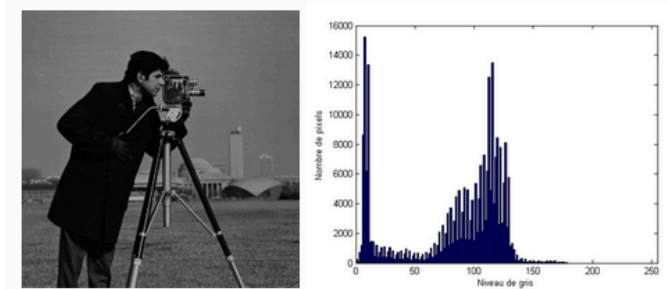


## 2. Histogramme et contraste

Lire la lumière d'une image

# Histogramme

- Représentation de la distribution des intensités de pixels
- Pour chaque intensité (0-255), on compte le nombre de pixels ayant cette intensité
- Permet de visualiser la luminosité de l'image
- Utilisé pour le seuillage



# Transformations d'histogramme

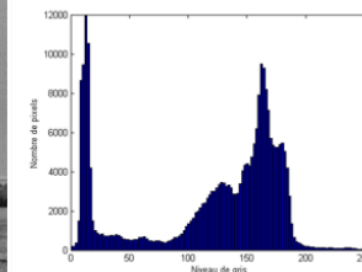
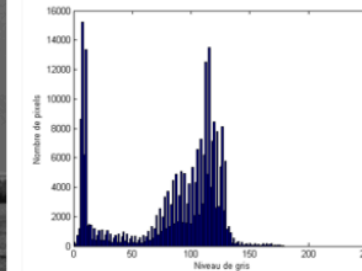
Les transformations d'histogramme permettent de **modifier la distribution** des intensités de l'image pour **améliorer le contraste et la luminosité**.

- **Normalisation** : L'ordonnée de l'histogramme est normalisée
  - → la norme (L1, L2, ...) est égale à 1.
  - Ne modifie pas l'image, seulement l'histogramme
- **Étirement** : transformation affine de l'histogramme entre deux valeurs
- **Égalisation** : Répartition uniforme des intensités

# Étirement

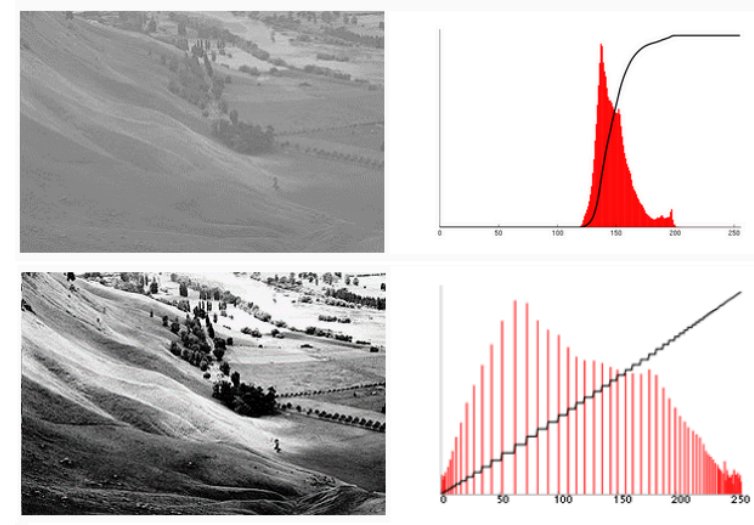
- Permet de corriger les variations d'éclairage, les ombres et les reflets
- **Augmente la dynamique** de l'image
- On cherche à **étirer** l'histogramme pour couvrir toute la plage de valeurs

Confusion fréquente avec la normalisation :  
`cv2.normalize` sert à effectuer l'étirement



# Égalisation d'histogramme

- Amélioration du contraste de l'image
- Mathématiquement, on cherche à obtenir une **distribution uniforme** des intensités

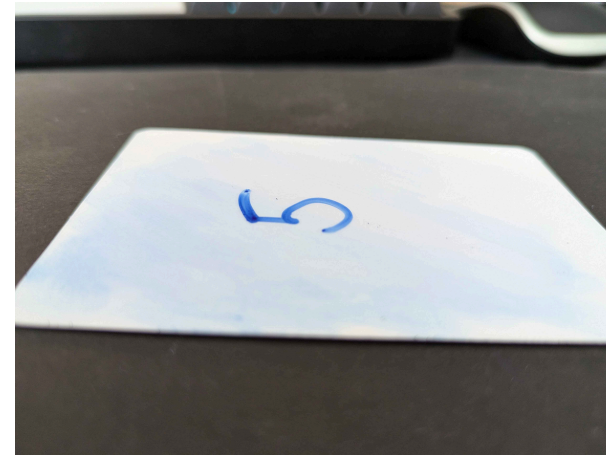


# Exercice n°2

Reprendre l'image de l'exercice n°1 après manipulation, puis :

- Étirer
- Égaliser

A chacune des 3 étapes, afficher l'image, l'histogramme normalisé (en probabilité) et l'histogramme cumulé



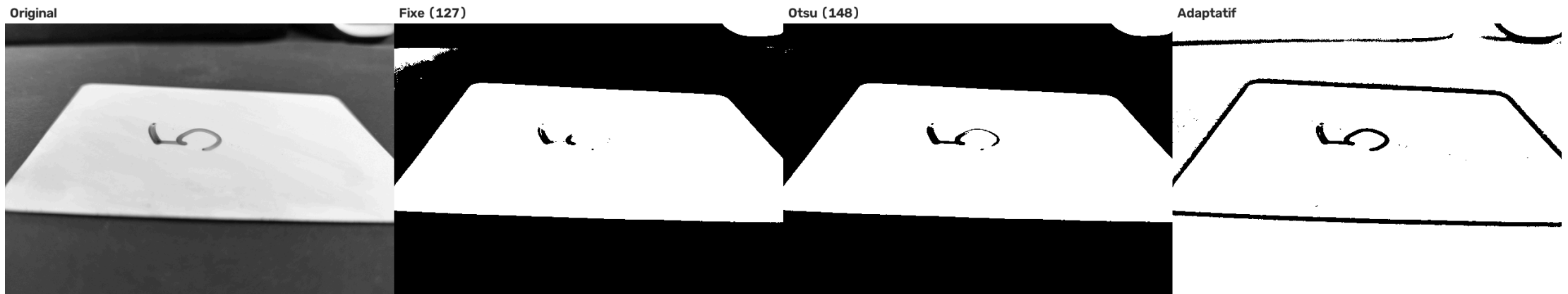


### 3. Segmentation par seuillage

Séparer l'objet du fond

# Seuillage simple

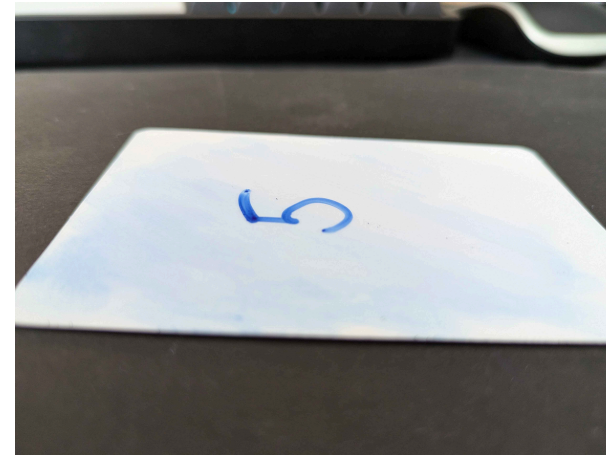
- Technique de segmentation d'image
- Permet d'isoler les objets d'intérêt
- On en sort une **image binaire** / un **masque** (0 ou 1)
- Un seuil unique pour toute l'image : fragile si l'éclairage varie



# Exercice n°3

Garder uniquement les pixels correspondant aux chiffres

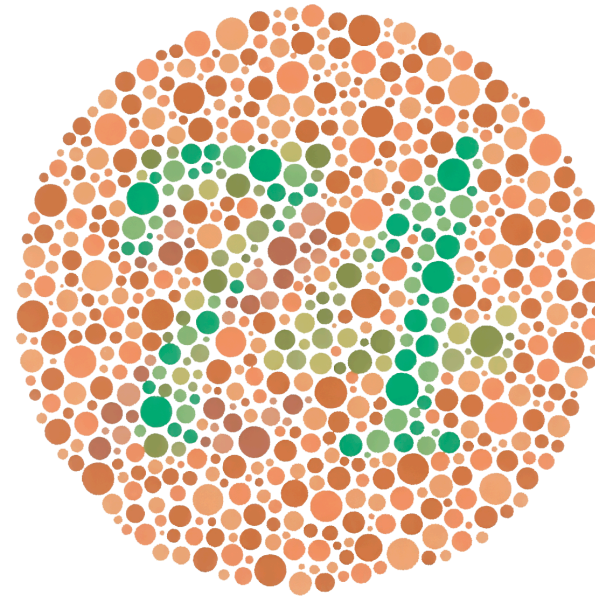
- Déterminer un seuil pour binariser l'image
- Afficher l'image seuillée



# Exercice n°3 bis

Garder uniquement les pixels correspondant aux chiffres

- Ouvrir & convertir l'image : RGB  $\rightarrow$  ?
- Afficher l'histogramme
- Choisir un seuillage sur l'image
- Afficher l'image seuillée



# Seuillage automatique

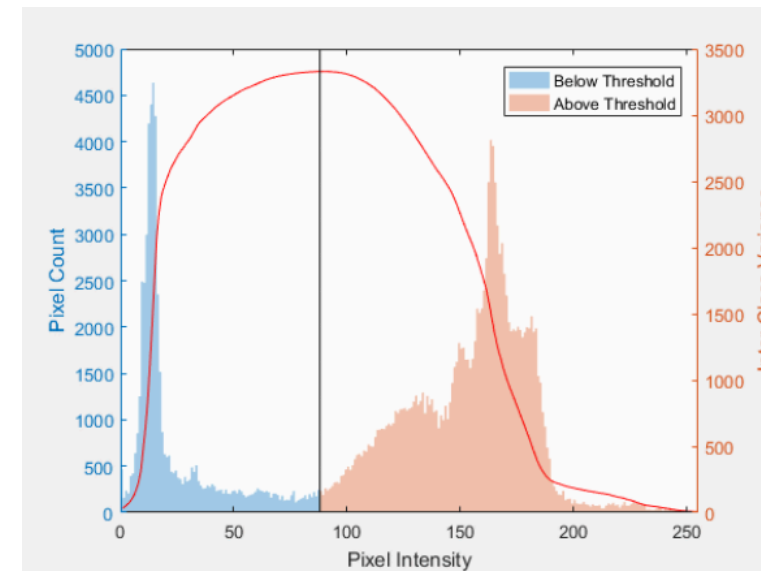
- Le seuil est calculé automatiquement à partir de l'histogramme de l'image
- Il reste **global** : un seul seuil pour toute l'image
- L'algorithme **OTSU** est la méthode la plus couramment utilisée
- Pour un éclairage variable, il faut un seuillage **adaptatif** (seuil local, voir plus loin)

# OTSU

- Trouver le seuil optimal pour binariser une image
- Minimise la variance intra-classe  $\equiv$  maximise la variance inter-classe
- On cherche à minimiser :

$$\sigma^2 = w_0 \times \sigma_0^2 + w_1 \times \sigma_1^2$$

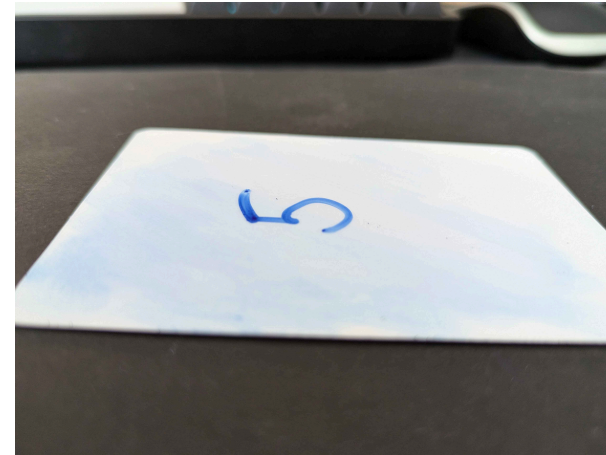
$w_0$  et  $w_1$  sont les probabilités d'appartenance à chaque classe.  $\sigma_0^2$  et  $\sigma_1^2$  sont les variances de chaque classe.



# Exercice n°4

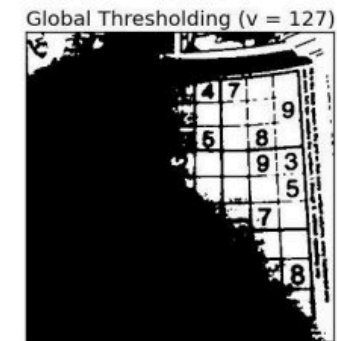
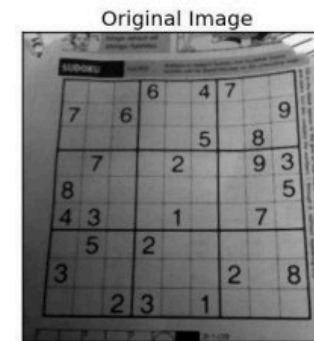
Sur l'image du premier exercice :

- Tester le seuillage automatique (OTSU)



# Seuillage adaptatif

Seuillage qui adapte le seuil en fonction de la région de l'image

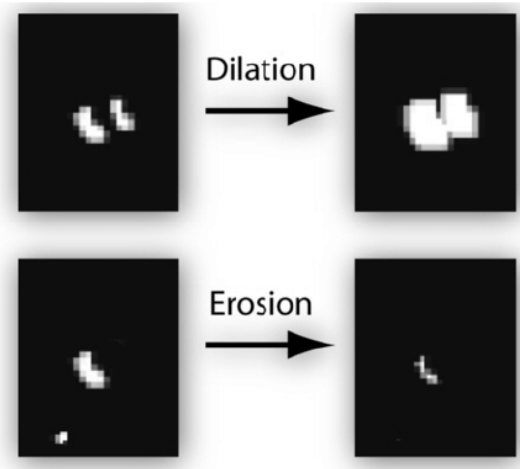


## 4. Morphologie

Nettoyer une image binaire

# Érosion, dilatation

- Opérations morphologiques sur les images binaires



- Noyau de convolution (3x3 pour l'exemple) :

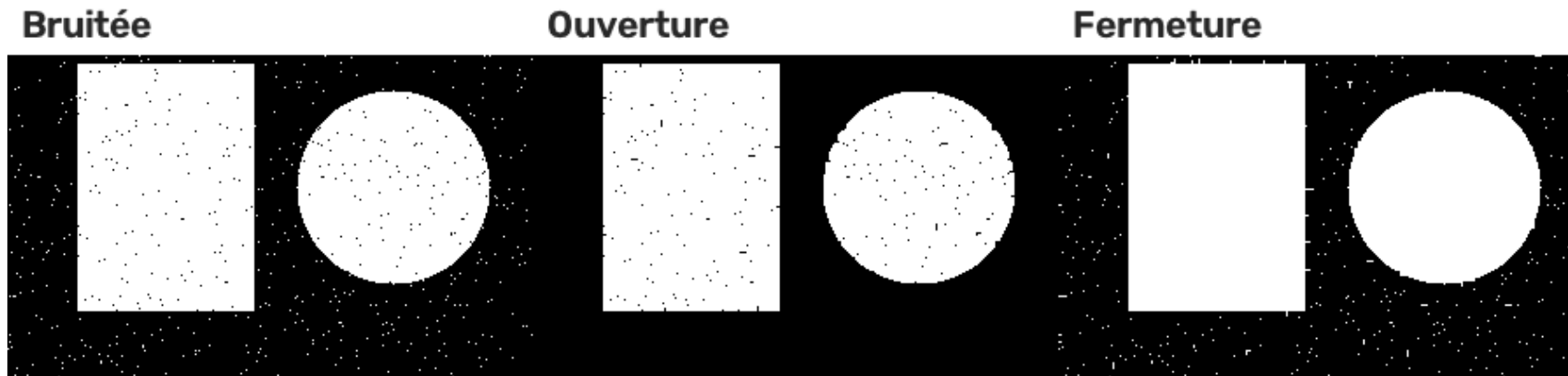
$$K = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

- Érosion** : "rétrécir" les objets
  - L'**un des pixels** couverts par le noyau est **noir**, le pixel central devient **noir**
- Dilatation** : "agrandir" les objets
  - L'**un des pixels** couverts par le noyau est **blanc**, le pixel central devient **blanc**

# Ouverture, fermeture

On veut **conserver la forme des objets** tout en nettoyant l'image binaire.

- **Ouverture** = **érosion puis dilatation** : supprime les petits objets parasites (bruit blanc)
- **Fermeture** = **dilatation puis érosion** : comble les petits trous dans les objets





## 5. Filtrage et contours

Lisser, puis trouver les bords

# Principaux filtres

- Les filtres passe-bas : lissage de l'image
- Les filtres passe-haut : détection de contours

# Les filtres passe bas

- **Filtre moyennneur** : plus rapide mais moins efficace
- **Filtre gaussien** : supprime le bruit gaussien
- **Filtre médian** : supprime le bruit impulsionnel (sel et poivre)

**Bruité**



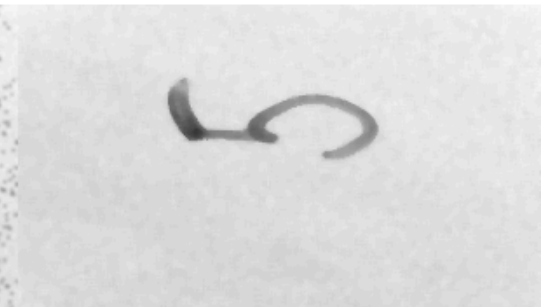
**Moyenneur 5x5**



**Gaussien 5x5**



**Médian 5x5**



# Les filtres passe haut

- **Detection de contours** : Sobel, Canny, & Co

Le principe est de mettre en évidence les variations d'intensité de l'image. Cette opération est réalisée en utilisant un noyau de convolution de ce type :

$$K = \begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$$

On accentue les différences d'intensité entre les pixels voisins

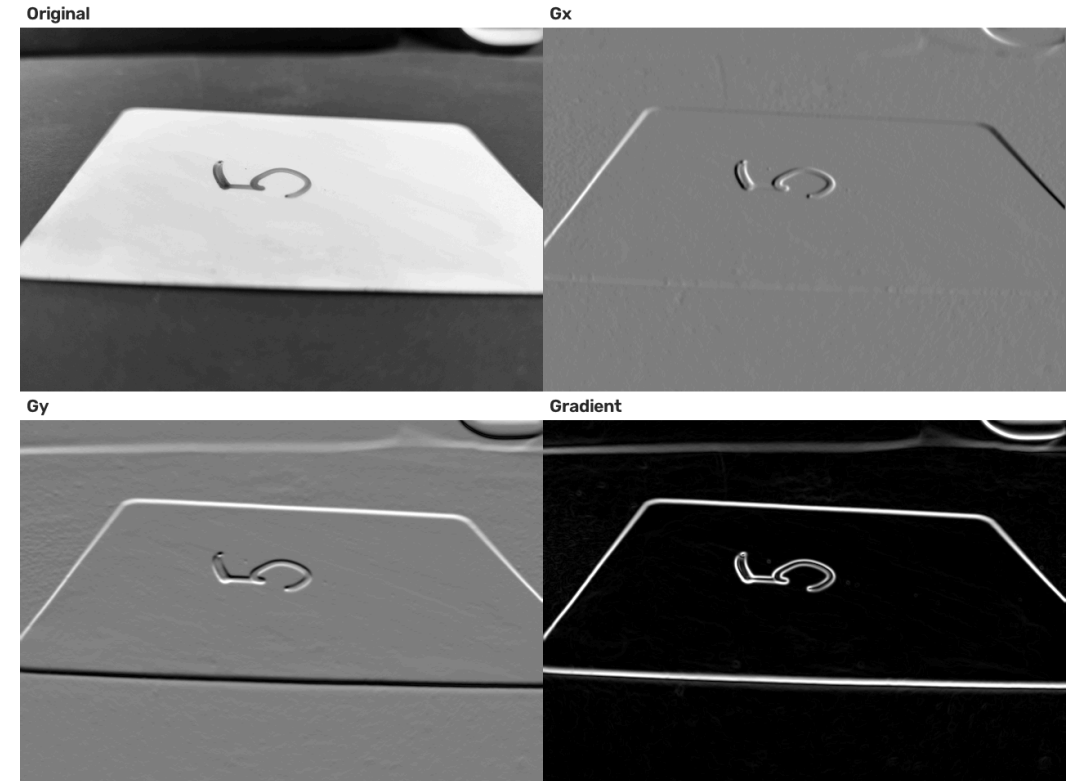
# Méthode de Sobel

Deux convolutions sont réalisées sur l'image pour obtenir les gradients horizontaux et verticaux.

$$G_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} * \text{image}$$

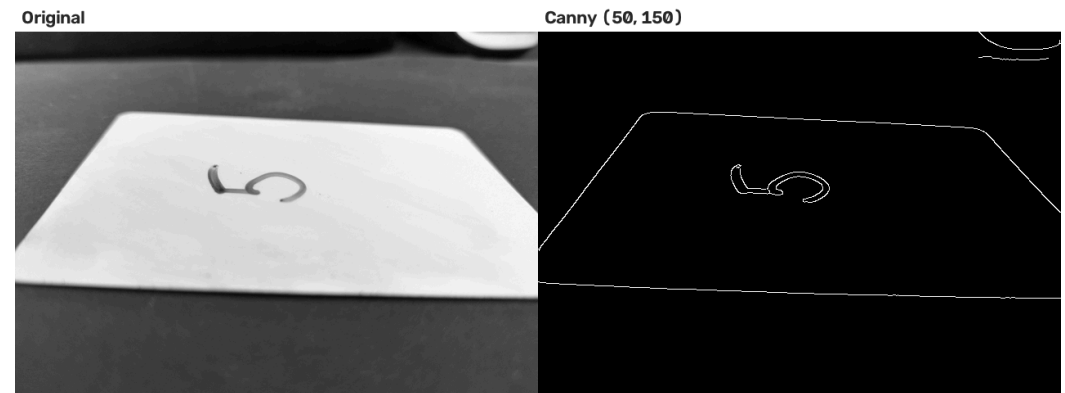
$$G_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix} * \text{image}$$

$$\text{Gradient} = \sqrt{G_x^2 + G_y^2}$$



# Méthode de Canny

1. **Filtrage gaussien** : suppression du bruit
2. **Gradient** : calcul du gradient de l'image
3. **Suppression des non-maxima** : suppression des pixels qui ne sont pas des maxima locaux
4. **Seuillage par hystérésis** : seuillage de l'image pour obtenir des contours nets



# Exercice n°5

- Implémenter un filtre gaussien avec la fonction `cv2.filter2D`
- Réimplémenter le filtre de Sobel avec la fonction `cv2.filter2D`
- Appliquer ces filtres sur l'image complète du premier exercice

## 6. Vidéo

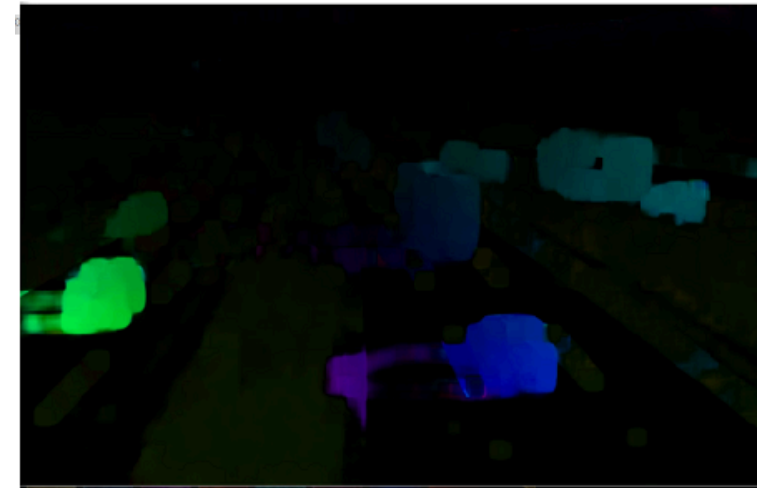
Quand le temps devient une information

# Traitements sur de la vidéo

- **Détection de mouvements** : différence d'images, flux optique
- **Soustraction d'arrière-plan** : extraction d'objets mobiles

# Flux optique

- Calcul du déplacement entre deux images consécutives
  - **Dense** : calcul du déplacement **pour chaque pixel** (Farneback)
  - **Sparse** : calcul du déplacement pour un **ensemble de points d'intérêt** (Lucas-Kanade)
- Utilisé pour la détection de mouvements, le suivi d'objets, la stabilisation d'image, navigation autonome, ...



# Exercice n°6

Calculer le flux optique de la vidéo `traffic.mp4` avec la méthode **Farneback**.

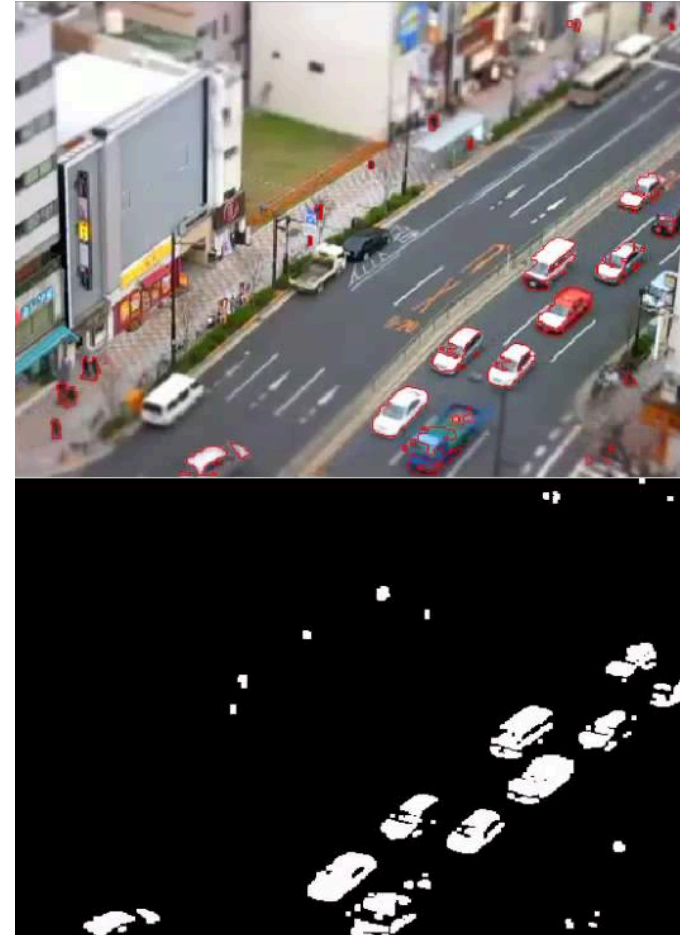


# Stabilisation d'image

- Corriger les mouvements indésirables de la caméra
  - Utilisé pour les vidéos prises à la main, les vidéos de surveillance, les vidéos aériennes, ...
  - **Méthode** : calcul du flux optique, calcul de la transformation affine, application de la transformation
- ! Problèmes : perte de qualité de l'image, déformation de l'image, zones noires

# Soustraction d'arrière-plan

- Extraction d'objets mobiles dans une séquence d'images
- Permet de détecter les objets en mouvement
- Utilisé pour la surveillance, la détection d'intrusion, le suivi d'objets
- Algorithme de soustraction d'arrière-plan : MOG, MOG2, KNN, GMG, ...



# Exercice n°7

Soustraire l'arrière-plan de la vidéo `traffic.mp4` avec la méthode **MOG2**.



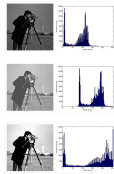
# Mise en garde

- Attention aux seuils fixes : ils cassent dès que l'éclairage change
- Préférer un seuillage automatique ou adaptatif, ou des méthodes sans seuillage

# À retenir

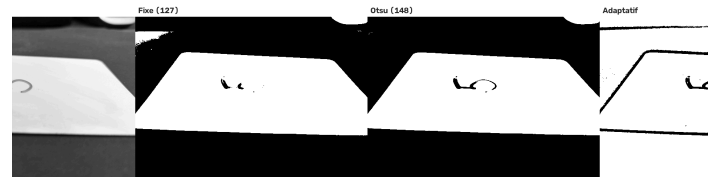
## 1. Regarder avant de corriger

L'histogramme dit si le contraste est le problème : étirer ou égaliser seulement alors.



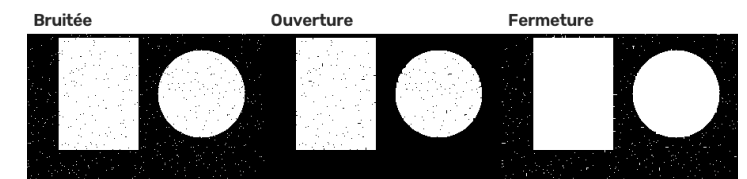
## 2. Un réglage fixe ne survit pas au réel

Seuil fixe : cassé dès que la lumière change.  
Otsu : global automatique. Adaptatif : local.



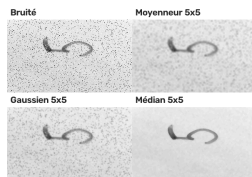
## 3. Nettoyer avant de mesurer

Ouverture : enlève le bruit. Fermeture : comble les trous.



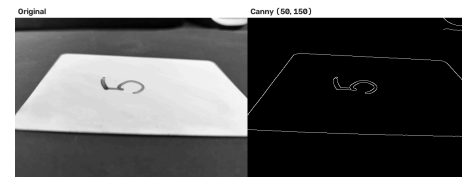
## 4. Le bon outil dépend du défaut

Gaussien pour le bruit gaussien, médian pour le sel et poivre.



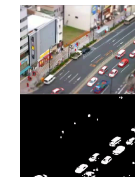
## 5. Dériver amplifie le bruit

Lisser avant Sobel ou Canny, sinon les contours se noient dans le bruit.



## 6. Le temps est une information

Comparer des images : flux optique pour le déplacement, MOG2 pour l'arrière-plan.



QUESTIONS ?