



الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'Enseignement Supérieure et de la Recherche Scientifique

جامعة وهران 2 محمد بن أحمد
Université d'Oran 2 Mohamed Ben Ahmed

معهد الصيانة والأمن الصناعي
Institut de Maintenance et de Sécurité Industrielle

Département de Maintenance en Instrumentation

MEMOIRE DE FIN D'ETUDES

Pour l'obtention du diplôme de Master

Filière : Génie Industriel
Spécialité : Génie Industriel

Thème

**Débruitage adaptatif et reconnaissance d'objets sur
une image en utilisant l'outil MATLAB**

Présenté et soutenu publiquement par :

BENAHMED Akram Ayemen

&

HEROUAL Chakib

Nom et Prénom	Garde	Etablissement	Qualité
Djilali Nekkrouf	MCB	Univ-Oran2/IMSI	Président
Adda Neggaz Samir	MAA	Univ-Oran2/IMSI	Encadrant
Reguieg Yssaad Sadek	MAA	Univ-Oran2/IMSI	Examineur

Année 2024/ 2025

Remerciements

*Nous remercions, avant tout, **Allah** le Tout puissant pour nous avoir donné la volonté et la patience pour accomplir ce travail.*

*Nous exprimons également nos sincères remerciements à **notre encadreur** de mémoire, pour son encadrement et ses précieux conseils, ainsi, que pour ses encouragements indispensables à la réalisation de ce travail.*

*Nous remercions **nos parents**, qui nous ont soutenus tout le long de nos études et nos familles qui, de près ou de loin nous ont apporté leur soutien.*

Nous remercions, également, tous les collègues du parcours universitaire.

*Nos vifs remerciements vont également **aux membres du jury** pour l'intérêt qu'ils ont à notre recherche en acceptant d'examiner notre travail et de l'enrichir par leurs propositions.*

Enfin, Nous remercions infiniment toutes les personnes qui ont contribué, de près ou de loin, à la réalisation de ce travail.

Dédicaces

Nous dédions ce modeste travail à :

*Nos mères, sources de tendresse et d'amours pour leurs
soutiens tout le long de notre vie scolaire.*

*Nos pères, qui nous ont toujours soutenus et qui ont fait tout
possible pour nous aider.*

A mes amis les hommes

Mon frère et mon soeur, que nous aimons beaucoup.

BENAHMED Akram Aymen

Résumé

Le traitement d'images numériques joue un rôle essentiel dans des domaines variés tels que la médecine, la télédétection et la robotique. Les images étant souvent affectées par différents types de bruit, des techniques de filtrage et de débruitage, allant des méthodes classiques aux approches adaptatives, ont été développées pour en améliorer la qualité. Parallèlement, la reconnaissance d'objets, renforcée par l'apprentissage profond, permet d'atteindre une grande précision dans des applications comme la conduite autonome et le diagnostic médical. Ce mémoire combine ainsi le débruitage adaptatif et la reconnaissance d'objets pour traiter efficacement des images bruitées et complexes.

Mots clés : Bruit d'image, Filtre d'image, détection d'objet.

Abstract

Digital image processing plays a vital role in diverse fields such as medicine, remote sensing, and robotics. Since images are often affected by various types of noise, filtering and denoising techniques, ranging from classical methods to adaptive approaches, have been developed to improve their quality. Meanwhile, object recognition, enhanced by deep learning, enables high accuracy in applications such as autonomous driving and medical diagnosis. This thesis combines adaptive denoising and object recognition to efficiently process noisy and complex images.

Keywords : Image noise, image filter, object detection.

ملخص

تلعب معالجة الصور الرقمية دورًا حيويًا في مجالات متنوعة، مثل الطب والاستشعار عن بُعد والروبوتات. ولأن الصور غالبًا ما تتأثر بأنواع مختلفة من الضوضاء، فقد طُوِّرت تقنيات تصفية وإزالة الضوضاء، بدءًا من الأساليب التقليدية وصولًا إلى الأساليب التكيفية، لتحسين جودتها. وفي الوقت نفسه، يُمكن التعرف على الأشياء، المُعزَّز بالتعلم العميق، من تحقيق دقة عالية في تطبيقات مثل القيادة الذاتية والتشخيص الطبي. تجمع هذه الأطروحة بين إزالة الضوضاء التكيفية والتعرف على الأشياء لمعالجة الصور المعقدة والضوضائية بكفاءة. الكلمات المفتاحية: ضوضاء الصورة، مرشح الصورة، كشف الأشياء.

Table des matières

Introduction générale	1
-----------------------------	---

CHAPITRE I : Définition et types d'image

I.1 Introduction	2
I.2 Définition d'image	2
I.3 Les principaux types d'images.....	2
I.3.1 Selon la nature des données types	3
I.3.1.1 Images binaires	3
I.3.1.2 Images en niveaux de gris	3
I.3.1.3 Images en couleur.....	3
I.3.2 Selon la dimension.....	5
I.3.3 Selon le format et la compression	6
I.3.3.1 Formats sans perte	6
I.3.3.2 Formats avec perte	6
I.3.4. Selon l'application.....	6
I.4 Les caractéristiques d'image.....	7
I.4.1 Pixel	7
I.4.2 Dimension	7
I.4.3 Résolution et Définition	8
I.4.4 La luminance	11
I.4.5 Contraste	8
I.5 Histogramme	9
I.5.1 Types d'histogrammes.....	9
I.5.1.1 Image en niveaux de gris.....	9
I.5.1.2 Image en couleur (RVB)	10
I.6 Le contour	11
I.6.1 Exemple filtre de cany et sobel	11
I.6.1.1 Méthode de Sobel.....	12
I.6.1.2 Méthode de Canny.....	12
I.7 Bruit	13
I.7.1 Bruit lié aux dispositifs d'acquisition.....	14

I.7.2 Bruit lié à l'échantillonnage	14
I.7.3 Bruit lié à la scène	14
I.7.4 Bruit lié à la transmodulation	14

Chapitre II : Les filtres adaptatifs et non adaptatifs

I.8 Conclusion	15
II.1 Introduction	16
II.2 Qu'est-ce qu'un filtre ?	16
II.3 Fonctionnement d'un filtre en traitement d'image	16
II.4 Les types principaux de filtres en traitement d'image	16
II.4.1 Filtres linéaires.....	16
II.4.1.1 Filtre passe-bas (filtre lisseur)	16
II.4.1.2 Filtre passe-haut.....	17
II.4.1.3 Filtrage spatial linéaire.....	17
II.4.1.4 Filtrage fréquentiel	17
II.4.2 Filtres non linéaires	18
II.5 Exemples des filtres.....	18
II.6 Les types de bruit dans les images numériques	18
II.6.1 Bruit gaussien	18
II.6.2 Bruit impulsionnel (sel et poivre)	19
II.6.3 Bruit de Poisson (bruit photonique).....	20
II.6.4 Bruit speckle.....	20
II.7 Filtrage.....	20
II.8 Pourquoi appliquer un filtrage sur une image ?	20
II.9 Les types des filtres	21
II.9.1 Filtre non adaptatif	21
II.9.1.1 Le filtrage moyennneur	21
II.9.1.2 Filtre gaussien	22
II.9.1.3 Le filtrage médian simple.....	24
II.9.2 Filtre adaptatif.....	25
II.9.2.1 Filtre de Wiener.....	25
II.9.2.2 Filtre médian	26

II.9.2.3 Filtre bilatéral.....	27
II.10 Comparaison pratique entre filtre moyenneur et médian	29
II.10.1 Avec bruit impulsif sel & poivre.....	29
II.10.1.1 Observation	29
II.10.2 Avec bruit gaussien.....	30
II.10.2.1 Observation	30
II.11 La dégradation des images	30
II.12 Conclusion.....	31

Chapitre II : Les filtres adaptatifs et non adaptatifs

III.1 Introduction	32
III.2 Introduction à la reconnaissance d'objets	32
III.3 Historique et évolution des techniques	32
III.4 Détection vs Reconnaissance : Clarification des Concepts.....	33
III.5 Structure pour effectuer la classification des images	33
III.6 Segmentation.....	34
III.6.1 Définition et objectifs.....	34
III.6.2 Principales approches de segmentation.....	35
III.6.2.1 Segmentation par contours.....	35
III.6.2.2 Segmentation par régions	35
III.6.2.3 Segmentation hybride.....	36
III.6.2.4 Segmentation basée sur l'apprentissage.....	36
III.6.3 Techniques et méthodes	36
III.6.3.1 Clustering (k-means, fuzzy c-means)	36
III.6.3.2 Seuillage	40
III.6.3.3 Méthodes morphologiques	41
III.6.4 Domaines d'application principaux de la segmentation d'images	44
III.7 Type de classification.....	44
III.7.1 Classification supervisée de l'image	44
III.7.2 Classification non supervisée de l'image	45
III.8 détection et reconnaissance d'objet	45
III.8.1 R-CNN (Region-based Convolutional Neural Network)	46

III.8.3 Faster R-CNN	47
III.8.4 Mask R-CNN	48
III.8.5 Single Shot MultiBox Detector	48
III.8.6 YOLO : You Only Look Once	49
III.8.6.1 Le modèle YOLOv2	51
III.8.6.2 Le modèle YOLOv3	51
III.8.6.3 Le modèle YOLOv4	52
III.9 Conclusion	52

Chapitre IV : Implémentation et Discussion des résultats

IV.1 Introduction	53
IV.2 Les images utilisées	53
IV.3 RGB to GRAY.....	55
IV.3.1 Application MATLAB (exemple perroquet)	55
IV.3.2 Résultat obtenu	55
IV.4 Histogramme d'image	56
IV.4.1 Résultat obtenu	56
IV.4.2 Description de la fonction 'imhist'	57
IV.4.3 Analyse des résultats	57
IV.4.3.1 Implémentation et paramètres techniques.....	57
IV.4.3.2 Interprétation des résultats obtenus	57
IV.5 Les méthodes de bruitage	57
IV.5.1 Bruit gaussien	57
IV.5.1.1 Script utilisé.....	58
IV.5.1.2 Résultat obtenu.....	58
IV.5.1.3 Discussion de résultat	58
IV.5.2 Bruit Spackle.....	59
IV.5.2.1 Script utilisé.....	59
IV.5.2.2 Résultat obtenu.....	59
IV.5.2.3 Discussion de résultat	60
IV.5.3 Bruit de Sel et Poivre (Salt and Pepper)	60
IV.5.3.1 Script utilisé.....	60

IV.5.3.2 Résultat obtenu.....	60
IV.5.3.3 Discussion de résultat	61
IV.5.4 Flou de mouvement (Motion Blur)	61
IV.5.4.1 Script utilisé.....	61
IV.5.4.2 Résultat obtenu.....	61
IV.5.5 Bruit de poisson (bruit photonique)	62
IV.5.5.1 Scripte utilisé.....	62
IV.5.5.2 Affichage de résultat.....	62
IV.6 Différence entre bruit gaussien et bruit speckle.....	63
IV.6.1 Distribution spatiale	63
IV.6.2 Effet visuel	63
IV.6.3 Méthodes de traitement.....	63
IV.7 Les filtre adaptatif	63
IV.7.1 Filtre de Wiener	63
IV.7.1.1 Scripte utilisé.....	64
IV.7.1.2 Affichage de figure.....	65
IV.7.1.3 Analyse des résultats.....	66
IV.7.2 Filtre Médian	66
IV.7.2.1 Script utilisé.....	66
IV.7.2.2 Résultat obtenu.....	67
IV.7.2.3 Analyse des résultats.....	67
IV.7.2.4 Implications pratiques et recommandations	68
IV.7.3 Filtre bilatéral	68
IV.7.3.1 Scripte utilisé.....	68
IV.7.3.2 Résultat obtenu.....	69
IV.7.3.3 Analyse des résultats.....	69
IV.8 Filtre non adaptatif.....	69
IV.8.1 Filtre moyennneur	69
IV.8.1.1 Scripte utilisé.....	69
IV.8.1.2 Résultat optenu.....	70
IV.8.1.3 Analyse des résultats.....	70
IV.8.2 Filtre de gaussien.....	70

IV.8.2.1 Scripte utilisé	70
IV.8.2.2 Résultat obtenu.....	71
IV.8.2.3 Analyse des résultats.....	71
IV.9 Détection et Reconnaissance d'objet	72
IV.9.1 YOLOv3.....	72
IV.9.1.1 Analyse des résultats.....	72
IV.9.2 YOLOv4.....	73
IV.9.2.1 Analyse des résultats.....	73
IV.9.3 Comparaison entre les deux model.....	73
IV.9.4 WEBCAM	74
IV.9.5.1 Qualité des detections.....	75
IV.9.5.2 Analyse le résultat	75
VI.10 Conclusion	75
Conclusion générale	76

Introduction générale

Le traitement d'images numériques occupe une place prépondérante dans de nombreux domaines scientifiques et technologiques, allant de la médecine à la télédétection, en passant par l'industrie, la robotique et les systèmes de sécurité. Avec l'essor des capteurs et de la puissance de calcul, la quantité d'images générées et à analyser ne cesse d'augmenter, rendant indispensable le développement de méthodes efficaces pour améliorer leur qualité et en extraire des informations pertinentes.

Cependant, les images sont souvent dégradées par différents types de bruit lors de leur acquisition, leur transmission ou leur stockage, ce qui complique leur interprétation. Pour remédier à ces limitations, diverses techniques de filtrage et de débruitage ont été développées, allant des filtres classiques non adaptatifs aux filtres adaptatifs plus avancés. Ces méthodes visent non seulement à réduire le bruit, mais aussi à préserver les contours et les détails essentiels de l'image.

Parallèlement, la reconnaissance d'objets est devenue un enjeu majeur en vision par ordinateur. Elle repose sur des étapes fondamentales telles que la segmentation, l'extraction de caractéristiques et la classification. L'intégration de techniques d'intelligence artificielle et d'apprentissage profond a permis d'améliorer significativement la précision et la rapidité de ces systèmes, ouvrant la voie à de multiples applications telles que la conduite autonome, la surveillance intelligente ou le diagnostic médical assisté par ordinateur.

Ce mémoire s'inscrit donc dans cette perspective, en combinant l'étude du débruitage adaptatif et des méthodes de reconnaissance d'objets, afin de proposer des solutions adaptées aux images bruitées et complexes.

CHAPITRE I :

Définition et types d'image

I.1 Introduction

L'image occupe une place centrale dans la société moderne, que ce soit dans la communication, l'art, la science ou la technologie. Avec l'avènement du numérique, la notion d'image s'est élargie pour englober des représentations visuelles variées telles que les photographies, les dessins, les images médicales ou satellitaires, toutes codées sous forme de matrices de pixels. Chaque pixel, élément fondamental de l'image numérique, encode des informations sur la couleur ou la luminosité, permettant ainsi de restituer fidèlement la scène observée. La compréhension des caractéristiques de l'image – dimensions, résolution, luminance, contraste, histogramme, bruit – est essentielle pour maîtriser les techniques de traitement et d'analyse, qui visent à améliorer la qualité visuelle, extraire des informations pertinentes ou automatiser des tâches complexes. Ce chapitre pose les bases nécessaires à l'étude approfondie du traitement d'images, en présentant les différents types d'images, leurs propriétés fondamentales et les principaux défis liés à leur acquisition et manipulation.

I.2 Définition d'image

L'image, représentation visuelle aux multiples formes (photographies, peintures, dessins ou images numériques), joue un rôle central dans la communication, l'art et les technologies modernes. D'un point de vue technique, une image numérique se matérialise sous la forme d'une matrice de $N \times M$ pixels, éléments de base obtenus par échantillonnage et quantification d'un signal capturé par un capteur. Chaque pixel, localisé par ses coordonnées (x, y) , x, y : coordonnées cartésiennes d'un point de l'image.

Encode un niveau de luminosité (niveau de gris) ou des composantes colorimétriques sur N bits, reflétant ainsi la luminosité ou la couleur de la scène observée. Cette structure de données, bien qu'abstraite, devient une information visuelle signifiante pour l'œil humain après affichage, permettant de saisir des émotions, documenter des réalités ou véhiculer des idées. Avec l'essor des médias sociaux, de la réalité virtuelle et des systèmes d'acquisition automatisés, ces matrices de pixels transcendent leur nature mathématique pour devenir un langage universel, à la croisée de la science des données et de la perception visuelle. [30]

I.3 Les principaux types d'images

Les images peuvent être classées selon plusieurs critères : la nature des données, la couleur, la dimension, ou encore leur format et application.

I.3.1 Selon la nature des données types

I.3.1.1 Images binaires

Une image binaire est une image numérique composée uniquement de deux valeurs possibles pour chaque pixel, généralement représentées par le noir et le blanc. Chaque pixel peut donc prendre la valeur 0 ou 1, où 0 correspond souvent à la couleur noire (arrière-plan) et 1 à la couleur blanche (objet) ou inversement selon la convention utilisée. Cette simplicité fait des images binaires les types d'images les plus élémentaires en traitement d'image, codées sur un seul bit par pixel. [31]

I.3.1.2 Images en niveaux de gris

La notion de niveaux de gris est l'échelle de valeurs dénotant diverses intensités de la lumière et ne contenant pas de couleur de dimensions ($M \times N$). En fonction du nombre de niveaux de gris possibles, la réponse à la question de savoir si des dégradés subtils apparaissent, est déterminée. Traditionnellement, il est maximisé à 256, à partir de 0 l'absence de lumière (noir) et à la lumière maximale (blanc). [1]

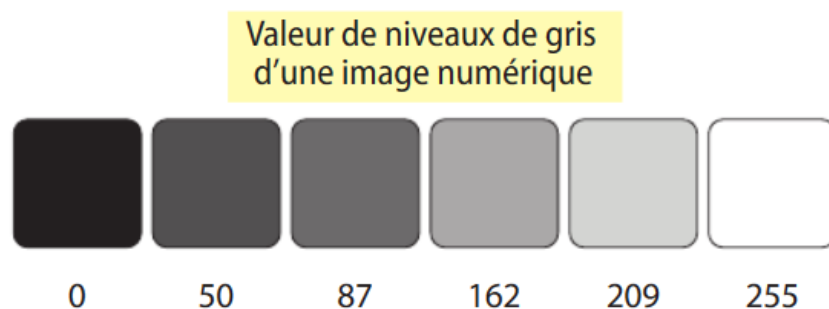


Figure I.1 : traitement d'image aux niveaux de gris

I.3.1.3 Images en couleur

I.3.1.3.1 Image en couleur RGB

Une image couleur représente des données multispectrales acquises par le domaine visible. Pour les images couleur classiques, ce domaine est réduit à trois canaux, un pour chaque couleur primaire : rouge, vert et bleu. Une façon simple de traiter ces images est d'utiliser le même algorithme pour chaque canal. [2]

En utilisant le modèle RGB (Rouge, Vert, Bleu) avec un codage de 8 bits par bande, chaque composante de couleur peut représenter 256 nuances différentes (de 0 à 255). Chaque pixel nécessite donc un total de 24 bits de stockage (3×8 bits), permettant de générer par :

$$2^{24} = 256 \times 256 \times 256 = 16\,777\,216 \text{ couleurs distinctes.}$$

Cette profondeur de couleur est communément appelée "couleurs vraies", car elle approche la capacité de l'œil humain à distinguer les nuances chromatiques. [4]

Pour des applications plus exigeantes comme la télévision haute définition (**HDR**), un codage de 10 bits par bande est souvent utilisé. Cette augmentation de la profondeur de bits permet d'obtenir 1024 nuances par composante (2^{10}), soit un total de :

$1024 \times 1024 \times 1024 = 1\,073\,741\,824$ couleurs possibles, C'est-à-dire environ 10,7 milliards de couleurs. Cette gamme étendue assure une reproduction plus fidèle des dégradés subtils et améliore significativement le rendu des zones d'ombre et de lumière.

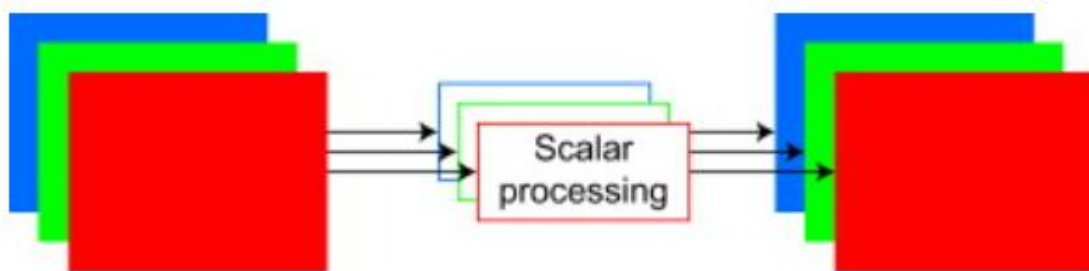


Figure I.2 : traitement d'images couleur

I.3.1.3.2 Image en couleur HSV

L'espace colorimétrique HSV (Hue, Saturation, Value) est un modèle cylindrique conçu pour représenter les couleurs de manière plus intuitive que le modèle RGB. Il décompose la couleur en trois composantes : la teinte (H) exprimée en degrés, la saturation (S) représentant la pureté, et la valeur (V) correspondant à la luminosité. Ce modèle est particulièrement adapté à la sélection de couleurs et au traitement d'images, notamment pour la segmentation ou la détection d'objets. Contrairement au RGB, HSV permet une manipulation indépendante de la couleur et de l'éclairage. Malgré son utilité, il ne reflète pas parfaitement la perception humaine, ce qui limite sa précision dans certaines applications colorimétriques. [42]

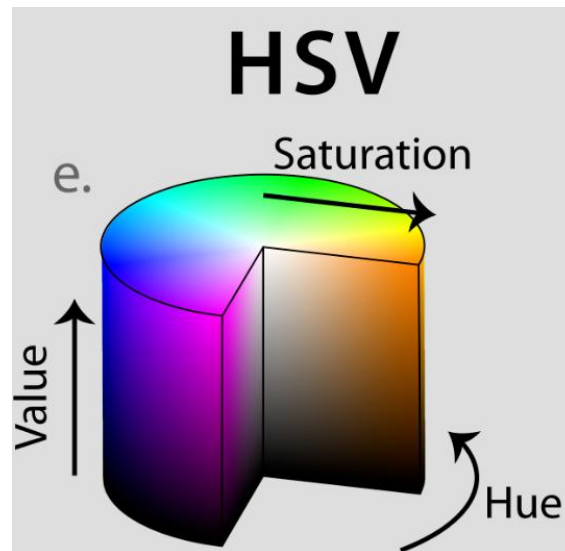


Figure I.3 : Visualisation du modèle de teinte, saturation et valeur (HSV) en 3D. [42]

I.3.2 Selon la dimension

Images 2D : Image représentée à l'écran dans un espace à deux dimensions (x, y), utilisées en photographie, imagerie médicale, ... etc. [37]

Images 3D : Utilisées en imagerie médicale (IRM, scanner) ou en vision par ordinateur pour la reconstruction 3D. [38]

Images hyper-spectrales : Les images hyper-spectrales sont un type d'imagerie qui capture et traite des informations sur l'ensemble du spectre électromagnétique pour chaque pixel d'une image. Contrairement aux images traditionnelles qui capturent seulement trois bandes spectrales (rouge, vert, bleu), ou aux images multispectrales qui en capturent généralement entre 4 et 10, les images hyper-spectrales peuvent enregistrer des centaines de bandes spectrales contiguës et étroites.

Chaque pixel d'une image hyper-spectrale contient donc un spectre complet, ce qui permet d'identifier avec précision les matériaux ou éléments présents dans la scène imagée. Cette technologie fournit beaucoup plus d'informations que la simple couleur visible, permettant de détecter des éléments invisibles à l'œil.

Les capteurs hyperspectraux divisent le spectre en bandes beaucoup plus nombreuses et plus étroites que les capteurs multispectraux, offrant ainsi une résolution spectrale bien supérieure. Cette richesse de données permet d'identifier avec précision la composition chimique des objets observés.

Cette technologie est utilisée dans de nombreux domaines comme la télédétection environnementale, l'agriculture de précision, la minéralogie, la surveillance militaire, la médecine, et l'inspection industrielle. [39]

I.3.3 Selon le format et la compression

Les formats de fichiers imagent les plus courants, et les plus importants selon le contexte d'utilisation (appareils photo, impression, numérisation, Internet), sont : **jpg**, **tif**, **png** et **gif**.

I.3.3.1 Formats sans perte

Conservent tous les détails, mais prennent plus de place.

tif (Tagged Image File) : y compris avec l'option de compression LZW, considéré comme le plus qualitatif pour les travaux professionnels, notamment dans le domaine de l'impression.

png (Portable Network Graphics) : conçu comme un remplaçant du GIF, il est aujourd'hui pris en charge par tous les navigateurs Web. Il propose des options plus avancées que le GIF, telles que la transparence, l'indexation ou l'utilisation de la couleur réelle (RVB), avec une profondeur de couleur allant de 1 à 48 bits. Il offre aussi certaines fonctionnalités du TIF.

I.3.3.2 Formats avec perte

Réduction de la taille avec perte de qualité.

jpg ou jpeg (Joint Photographique Expert Groupe) : utilisé lorsque la taille du fichier est plus importante que la qualité maximale de l'image, comme pour les pages Web, les courriels ou les cartes mémoire. Il s'agit d'un format avec perte, mais très efficace pour compresser les images. [29]

gif (Graphics Interchange Format) : créé par CompuServe à l'époque des premiers affichages 8 bits, avant l'arrivée du JPG. Il a été conçu pour l'affichage rapide d'images sur les connexions à bas débit (modem téléphonique). Il est limité à 256 couleurs mais permet les animations simples. [29]

I.3.4. Selon l'application

Images satellitaires : Capturées par des satellites pour la cartographie et la surveillance environnementale.

Images médicales : Rayons X, IRM, échographies utilisées en diagnostic médical.

Images thermiques : Capturent les variations de température, utilisées en vision nocturne et en surveillance.

I.4 Les caractéristiques d'image

I.4.1 Pixel

Un pixel, contraction de picture element, est l'unité de base d'une image numérique, représentant un point unique dans une matrice bidimensionnelle. Chaque pixel contient des informations sur sa couleur ou son intensité lumineuse, codées sous forme de valeurs numériques. Dans une image en niveaux de gris, un seul canal (généralement sur 8 bits) suffit pour définir 256 nuances, tandis qu'une image couleur utilise généralement trois canaux (RVB, pour Rouge, Vert, Bleu) codés sur 24 bits, permettant des millions de combinaisons chromatiques. La résolution d'une image dépend du nombre total de pixels (par exemple, 1920×1080 pour le Full HD), et leur densité (en pixels par pouce, PPP) influence la finesse des détails. Les pixels sont manipulés en traitement d'image pour des opérations comme le débruitage ou l'amélioration des contrastes, et leur codage varie selon les formats (JPEG, PNG, etc.). [40]

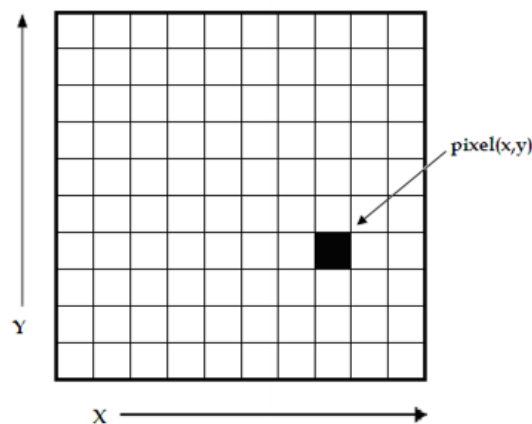


Figure I.4 : Exemple représentation matricielle d'une image numérique et localisation d'un pixel.

I.4.2 Dimension

La taille d'une image correspond aux dimensions de la matrice qui la représente numériquement. Cette matrice est constituée d'éléments appelés pixels, chacun traduisant une valeur d'intensité lumineuse. Ainsi, le produit du nombre de lignes par le nombre de colonnes de cette matrice donne le nombre total de pixels composant l'image. Cette structure matricielle constitue la base de toute opération en traitement d'image numérique. [43]

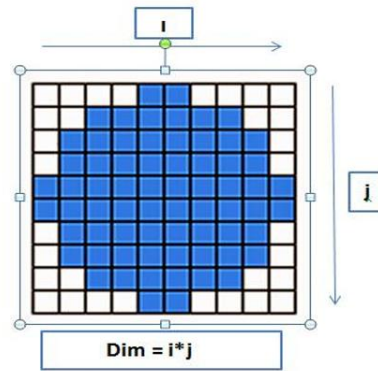


Figure I.5 : dimension d'image. [42]

I.4.3 Résolution et Définition

La qualité et la netteté d'une image numérique sont déterminées par deux notions fondamentales liées aux pixels : **la définition et la résolution**.

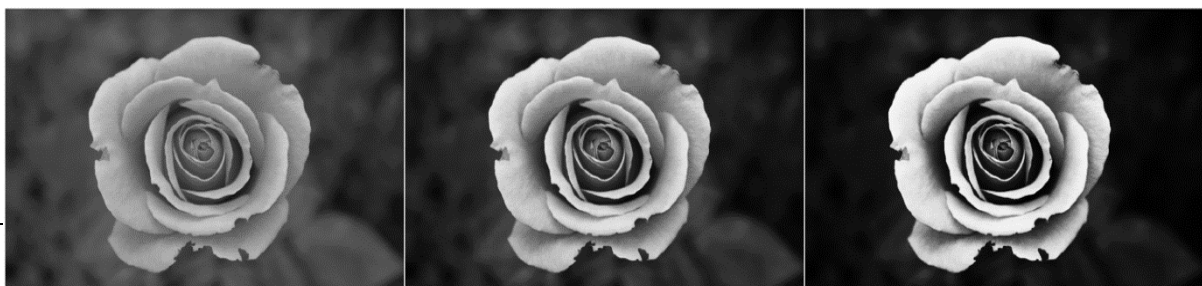
La définition représente le nombre total de pixels composant une image, exprimée sous forme de dimensions (largeur $\times$ hauteur). Par exemple, une image ayant une définition de 1500×1200 contient 1500 pixels en largeur et 1200 pixels en hauteur, soit un total de 1,8 million de pixels. Cette valeur correspond à la quantité brute d'information contenue dans l'image. [6]

La résolution quant à elle, mesure la densité de pixels par unité de longueur physique. Elle s'exprime généralement en PPP (points par pouce) ou DPI (dots per inch) en anglais, parfois aussi en points par centimètre. Une résolution de 400 DPI signifie que l'image comporte 400 pixels sur une longueur d'un pouce dans chaque direction, soit 160 000 pixels (400×400) sur un pouce carré. La résolution détermine donc comment les pixels seront répartis lors de l'impression ou de l'affichage. [6]

I.4.5 Contraste

Le contraste d'une image désigne la différence d'intensité lumineuse entre les zones claires et sombres, c'est la différence marquée entre deux régions d'une image, influençant la perception des détails. Il se mesure par l'écart entre les valeurs maximales et minimales de luminance ou par le rapport signal/bruit local. [2]

$$I(i, j) = a \cdot (I(i, j) - 128)$$



(a) (b) (c)

Figure I.7 : Effets de la variation de contraste d'une image en niveaux de gris. (a) Image à faible contraste. (b) image originale. (c) Image à forte contraste.

I.5 Histogramme

L'histogramme d'une image, qu'il s'agisse des niveaux de gris ou des couleurs, représente la distribution des intensités lumineuses en indiquant la fréquence d'apparition de chaque niveau dans l'image. Il constitue un outil essentiel pour l'analyse et le traitement d'image, car il permet d'évaluer la répartition de l'information visuelle, de détecter les déséquilibres d'exposition, et d'identifier les plages de niveaux dominants. En modifiant cet histogramme par des opérations comme l'égalisation ou l'étirement. On peut améliorer le contraste, corriger les effets de variation d'éclairage, ou réduire les erreurs de quantification. Ces ajustements facilitent notamment la comparaison entre plusieurs images acquises dans des conditions différentes ou permettent d'extraire des informations utiles qui seraient autrement masquées. Ainsi, la transformation de l'histogramme constitue une étape clé dans le rehaussement de la qualité d'image, en contribuant à une meilleure lisibilité et à une analyse plus fiable des contenus visuels. [42]

L'histogramme de couleur H d'une image I contenant N pixels et n couleurs peut être représenté sous forme de vecteur :

$$H = (h_{c1}, h_{c2}, \dots, h_{cn}) \quad [14]$$

I.5.1 Types d'histogrammes

I.5.1.1 Image en niveaux de gris

L'histogramme des niveaux de gris, aussi appelé histogramme d'intensité, est une représentation graphique de la distribution des niveaux de luminance dans une image en niveaux de gris. Il illustre la manière dont les valeurs d'intensité lumineuse sont réparties dans l'image.

Chaque niveau de gris correspond à une valeur d'intensité comprise entre 0 (noir) et 255 (blanc) dans une image codée sur 8 bits. L'histogramme associe à chaque niveau de luminance le nombre de pixels dans l'image possédant cette valeur. Ainsi, il permet d'analyser la dominance de certaines teintes de gris ainsi que le contraste global de l'image. [14]

L'étude de cet histogramme est essentielle dans de nombreuses applications de traitement d'images, notamment pour :

- L'amélioration du contraste (par égalisation ou étirement),
- Le seuillage pour la segmentation d'image,
- La détection de défauts visuels.

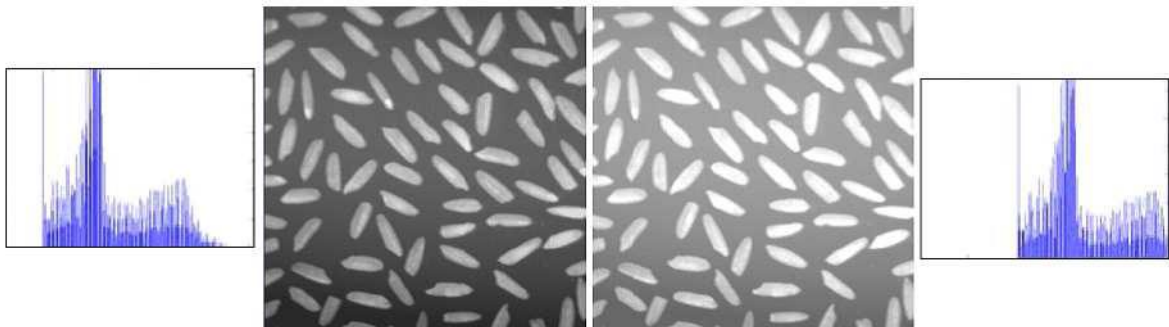


Figure I.8 : Comparaison de l'image 'rice.png' et de l'image améliorée avec leurs histogrammes respectifs

I.5.1.2 Image en couleur (RVB)

La majorité des descripteurs de couleur utilisés en traitement d'images sont basés sur des histogrammes de couleur, ou dérivent directement de ceux-ci. Ces descripteurs ont pour objectif de capturer la distribution des couleurs présentes dans une image ou dans une région spécifique de celle-ci. Le processus consiste d'abord à projeter l'image dans un espace couleur adapté (comme RGB, HSV ou CIELab), puis à quantifier les valeurs de couleur, et enfin à compter la fréquence d'apparition de chaque couleur quantifiée. [14]

Lorsqu'on travaille avec des distributions continues ou contenant un grand nombre de valeurs possibles, il devient nécessaire de regrouper les valeurs en intervalles ou "clusters", formant ainsi des classes de couleurs. Chaque bin de l'histogramme représente le nombre d'occurrences de couleurs appartenant à un intervalle donné. Par exemple, une image codée en 24 bits peut théoriquement contenir jusqu'à 2^{24} couleurs distinctes, soit plus de 16 millions. [14]

I.6 Le contour

Le contour d'une image est défini comme l'ensemble des pixels où l'on observe une variation significative de l'intensité lumineuse. Il marque les discontinuités dans la structure visuelle représentant les frontières entre différentes régions ou objets présents dans l'image. L'extraction de contours constitue une étape clé dans de nombreuses applications de vision par ordinateur, telles que la segmentation, la reconnaissance de formes et l'analyse d'objets.[3]

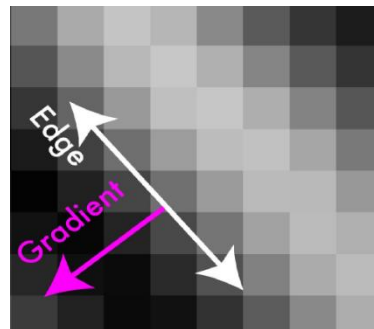


Figure I.9 : Illustration de la notion de gradient et de contour dans une image numérique

I.4.4 La luminance

C'est le degré de luminosité des points de l'image. Elle est définie aussi comme étant le quotient de l'intensité lumineuse d'une surface par l'aire apparente de cette surface, le mot luminance est substitué au mot brillance, qui correspond à l'éclat d'un objet [41]. Une bonne luminance se caractérise par :

- Des images lumineuses (brillantes)
- Un bon contraste : Il faut éviter les images où la gamme de contraste tend vers le blanc ou le noir ; ces images entraînent des pertes de détails dans les zones sombres ou lumineuses.
- L'absence de parasites.

Pour augmenter la luminance, il suffit de décaler l'histogramme :

$$I(i,j) = I(i,j) + b$$

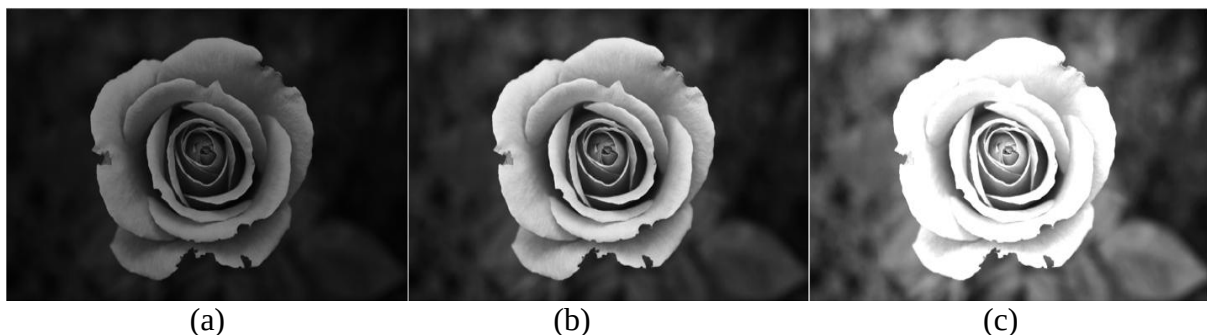


Figure I.6 : Effets de la variation de luminance d'une image en niveaux de gris. (a) Image à faible luminance. (b) image originale. (c) Image à forte luminance.

I.6.1 Exemple filtre de cany et sobel

I.6.1.1 Méthode de Sobel

La méthode de Sobel repose sur le calcul du gradient de l'intensité lumineuse de chaque pixel de l'image. Ce gradient permet d'identifier à la fois la direction dans laquelle l'intensité varie le plus fortement (du clair vers le foncé), ainsi que le taux de changement dans cette direction. Les zones présentant un changement brusque de luminosité correspondent généralement aux contours des objets présents dans l'image.

Pour ce faire, l'opérateur de Sobel utilise des masques de convolution, généralement de taille 3×3 , appliqués selon les directions horizontale et verticale. Ces masques permettent d'estimer les dérivées partielles de l'intensité, conduisant ainsi à la détection des bords et à la détermination de leur orientation. [12]

I.6.1.2 Méthode de Canny

Le filtre de Canny, ou détecteur de Canny, est une méthode avancée de détection de contours développée par John Canny en 1986. Son objectif est d'offrir une détection optimale selon trois critères fondamentaux :

1. **Bonne détection** : minimiser le taux d'erreur en assurant que tous les contours significatifs soient détectés.
2. **Bonne localisation** : les contours détectés doivent être aussi proches que possible des contours réels de l'image.
3. **Réponse unique** : chaque véritable contour doit générer une seule réponse, sans doublons ni faux positifs. [12]



Figure I.10 : Image ‘coins’

Après l'utilisation de canny et sobel edge detector :

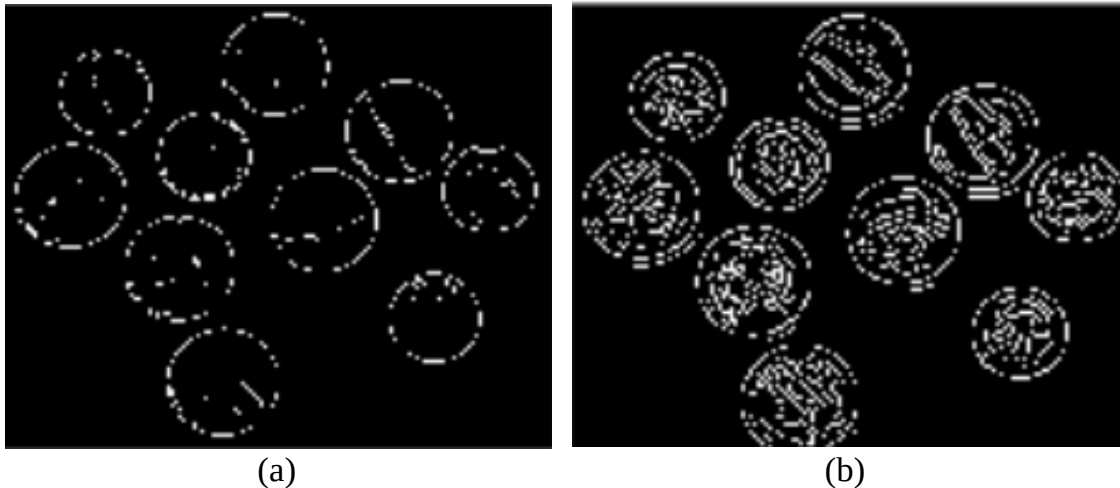


Figure I.11 : Détection des contours d'une image par les opérateurs de Sobel (a) et Canny (b)

I.7 Bruit

Le bruit dans le contexte du traitement d'images est défini comme une perturbation aléatoire superposée au signal original, altérant ainsi la qualité de l'image capturée. Il se manifeste par des variations imprévues dans les niveaux de gris ou les couleurs, ne correspondant à aucune information réelle de la scène observée. Le bruit est généralement modélisé par des processus stochastiques et caractérisé par ses propriétés statistiques (moyenne, variance, densité spectrale, etc.). [5]

Sa présence peut dégrader les performances de nombreux algorithmes de traitement d'image, notamment ceux liés à la détection et à la segmentation.

I.7.1 Bruit lié aux dispositifs d'acquisition

Ce type de bruit est directement lié à l'électronique des capteurs utilisés pour l'acquisition de l'image. Il comprend :

- Le bruit thermique généré par les composants électroniques, en particulier dans les circuits de lecture,
- Le bruit de lecture associé à l'amplification et à la conversion analogique-numérique,
- Le bruit quantique dû à la nature discrète de la lumière (photons).

Ces bruits sont souvent modélisés comme des bruits gaussiens blancs additifs et affectent de manière globale la qualité de l'image. [30]

I.7.2 Bruit lié à l'échantillonnage

Lors de l'acquisition d'une image, le signal continu de la scène est converti en signal discret via le processus d'échantillonnage spatial. Si ce processus est mal configuré (fréquence d'échantillonnage trop faible), il peut générer un phénomène de repliement spectral (aliasing), introduisant une forme de bruit dans l'image.

Ce bruit est typiquement structuré et peut se manifester sous forme de motifs indésirables, rendant certaines zones difficiles à analyser. [5]

I.7.3 Bruit lié à la scène

Ce bruit provient des conditions d'illumination ou des mouvements internes de la scène au moment de la capture. Par exemple :

- Des objets en mouvement pendant une pose longue peuvent entraîner du flou de mouvement.
- Des fluctuations de l'éclairage peuvent générer des artefacts d'intensité.

Bien que ce type de bruit ne soit pas directement lié au capteur, il introduit une variabilité non contrôlée dans l'image. [30]

I.7.4 Bruit lié à la transmodulation

La transmodulation regroupe les processus de conversion du signal entre différentes formes (par exemple, optique à électronique, ou analogique à numérique). Ces conversions peuvent introduire des distorsions ou du bruit supplémentaire, notamment à cause de :

- La non-linéarité des circuits,
- La quantification lors de la numérisation,
- Les erreurs de calibration ou de gain.

Ce bruit peut être difficile à modéliser précisément et nécessite souvent des approches adaptatives pour être corrigé efficacement. [5]

I.8 Conclusion

En résumé, ce chapitre a permis d'explorer les fondements de l'image numérique, en détaillant ses différentes formes, ses caractéristiques essentielles et les principaux types de bruit susceptibles d'altérer sa qualité. La compréhension de ces notions est indispensable pour aborder efficacement les techniques de traitement d'images, qu'il s'agisse de filtrage, de restauration ou d'extraction de caractéristiques. Les concepts de pixel, de résolution, de luminance et de contraste, ainsi que la connaissance des différents formats et types d'images, constituent le socle sur lequel reposent toutes les méthodes avancées de traitement et d'analyse. Cette base théorique ouvre la voie à l'étude des outils et algorithmes qui seront développés dans les chapitres suivants, en vue d'optimiser la qualité et l'exploitation des images numériques dans des contextes variés.

Chapitre II:

Les filtres adaptatifs

et

non adaptatifs

II.1 Introduction

Dans le domaine du traitement d'images numériques, le filtrage est une étape essentielle permettant d'améliorer la qualité des images, de réduire le bruit ou encore de détecter des contours. Les filtres sont des outils mathématiques appliqués sur les pixels d'une image pour transformer ses caractéristiques selon des objets précises.

Ce chapitre présente les deux grandes familles de filtres : les filtres linéaires et non linéaires, ainsi que la différence entre filtres adaptatifs et non adaptatifs. À travers des définitions, des exemples et des explications pratiques, nous découvrirons le rôle de chacun dans la réduction du bruit, la préservation des contours et l'amélioration visuelle des images.

II.2 Qu'est-ce qu'un filtre ?

En traitement d'image, un filtre est un opérateur mathématique appliqué à une image numérique pour en modifier les valeurs des pixels en fonction de celles de leurs voisins, dans le but d'extraire des informations, améliorer l'image ou en modifier certaines caractéristiques. Ce processus s'effectue généralement par une opération appelée convolution avec un noyau (ou masque) de convolution, qui est une petite matrice de coefficients appliquée localement sur l'image. [17]

II.3 Fonctionnement d'un filtre en traitement d'image

Le filtre agit sur chaque pixel en combinant sa valeur avec celles des pixels environnants, pondérées par les coefficients du noyau.

Le noyau est souvent une matrice carrée (par exemple 3x3, 5x5) dont le centre correspond au pixel traité.

Le résultat est une nouvelle image où les pixels ont été transformés selon le type de filtre appliqué.

II.4 Les types principaux de filtres en traitement d'image

II.4.1 Filtres linéaires

Ils réalisent une combinaison linéaire des pixels voisins via la convolution. Par exemples :

II.4.1.1 Filtre passe-bas (filtre lisseur)

Atténue les hautes fréquences, ce qui a pour effet de flouter l'image et de réduire le bruit. Par exemple, un filtre moyenneur ou un filtre gaussien. [17]

II.4.1.2 Filtre passe-haut

Accentue les hautes fréquences, permettant de faire ressortir les contours et détails (extraction de contours). Par exemple, les filtres de Sobel, Prewitt, ... ext. [17]

II.4.1.3 Filtrage spatial linéaire

Le filtrage spatial linéaire consiste essentiellement en une opération de convolution entre une fonction d'intensité et une fonction appelée réponse impulsionnelle du filtre.

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

$$g(x, y) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} h(u, v) f(x - u, y - v) du dv$$

Dans le cas direct :

$$g(x, y) = \sum_{u=-h/2}^{+h/2} \sum_{v=-h/2}^{+h/2} h(u, v) f(x - u, y - v)$$

Où correspond à la dimension de l'opérateur de filtrage. [28]

II.4.1.4 Filtrage fréquentiel

Le filtrage fréquentiel consiste à transformer l'image du domaine spatial vers le domaine fréquentiel, où le traitement est effectué, puis à appliquer la transformation inverse pour revenir au domaine spatial. Les transformations utilisées sont la Transformée de Fourier directe (TF) et sa transformée inverse, appliquées à l'image. Elles sont définies respectivement par les expressions suivantes :

$$F(u, v) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x, y) e^{-j(ux+vy)} dx dy$$

$$f(x, y) = \frac{1}{4\pi^2} \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} F(u, v) e^{+j(ux+vy)} du dv$$

Dans le cas discret, la Transformée de Fourier Discrète directe (TFD) et son inverse sont données respectivement par :

$$F(\Omega, \Psi) = \sum_{m=-\infty}^{+\infty} \sum_{n=-\infty}^{+\infty} f(m, n) e^{-j(\Omega m + \Psi n)}$$

$$f(m, n) = \frac{1}{4\pi^2} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} F(\Omega, \Psi) e^{j(\Omega m + \Psi n)} d\Omega d\Psi$$

Pour une image filtrée, la convolution dans le domaine spatial est équivalente à un produit dans le domaine fréquentiel et vice versa, on a donc :

$$\begin{cases} G = F(g) = F(h * f) = H \cdot F \\ g = F^{-1}(H \cdot F) \end{cases}$$

Où est l'opérateur de la TF. [28]

II.4.2 Filtres non linéaires

On appelle filtre non linéaire un opérateur qui remplace la valeur d'un pixel par une combinaison non linéaire des valeurs de ses pixels voisins. L'objectif principal des filtres lisseurs non linéaires est de réduire le bruit tout en préservant les détails importants de l'image, notamment les contours et les structures fines, ce que les filtres linéaires peinent souvent à faire efficacement. [18]

II.5 Exemples des filtres

Filtre moyeneur : remplace chaque pixel par la moyenne des pixels voisins, ce qui lisse l'image. [43]

Filtre gaussien : applique un flou basé sur la distribution gaussienne, très utilisé pour réduire le bruit tout en préservant les contours mieux que le filtre moyeneur. [17]

Filtre de Sobel : détecte les contours en calculant des dérivées premières dans les directions horizontale et verticale. [28]

Filtre de Canny : méthode avancée de détection de contours combinant un flou gaussien, un filtre de dérivée, et un seuillage optimisé. [28]

II.6 Les types de bruit dans les images numériques

La nature du bruit présent dans une image dépend fortement du système d'acquisition, des conditions d'environnement et du type de scène. Plusieurs modèles statistiques sont utilisés pour représenter les bruits les plus fréquemment rencontrés. Ces modèles permettent de concevoir des méthodes de filtrage adaptées à chaque situation. [2]

II.6.1 Bruit gaussien

Le bruit gaussien (ou bruit normal) est le plus couramment rencontré dans les systèmes d'imagerie. Il est généralement causé par les fluctuations thermiques des capteurs

électroniques et les erreurs de lecture du signal. Il est caractérisé par une distribution normale de probabilité, de moyenne nulle et de variance constante :

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\left(\frac{(x-\mu)^2}{2\sigma^2}\right)} \quad [71]$$

Où

- $p(x)$, on note que la gamme des valeurs de x où $-\infty < x < +\infty$
- x représente l'intensité
- μ est la moyenne de z .
- σ est son écart-type.
- L'écart-type au carré σ^2 , s'appelle la variance de x .

Ce type de bruit est additif, ce qui signifie qu'il se superpose au signal original sans le modifier structurellement. Les valeurs doivent être non négatives. [5]

II.6.2 Bruit impulsionnel (sel et poivre)

Le bruit impulsionnel, également appelé bruit "sel et poivre", est un type de bruit aléatoire qui affecte les images numériques. Il se manifeste par des pixels qui prennent de manière aléatoire les valeurs extrêmes de l'échelle des niveaux de gris, à savoir 0 (noir) ou 255 (blanc), donnant ainsi l'apparence de grains de sel et de poivre éparpillés sur l'image.

Ce type de bruit peut survenir lors de la numérisation d'une image ou au cours de sa transmission, en raison d'erreurs de capteurs ou de perturbations électroniques. Il est particulièrement destructeur car il altère directement certains pixels sans affecter leurs voisins, ce qui rend sa suppression complexe. [22]

Le bruit sel et poivre peut être modélisé mathématiquement comme suit :

$$I(t) = (1 - e) \cdot S(t) + e \cdot N(t)$$

Où :

- $I(t)$ Le signal (l'image) à l'instant.
- $S(t)$ est l'image originale.
- $N(t)$ est le bruit impulsionnel.
- e est le taux de bruit (entre 0 et 1), exprimant la proportion de pixels affectés. [22]

II.6.3 Bruit de Poisson (bruit photonique)

Le bruit photonique est une forme de bruit inhérente à la nature aléatoire de la lumière. Il résulte des fluctuations statistiques du nombre de photons captés par le capteur d'un appareil photo. Ce bruit est particulièrement perceptible dans des conditions de faible luminosité ou lorsque les réglages de l'appareil limitent la quantité de lumière reçue. [23]

II.6.4 Bruit speckle

Le bruit de speckle apparaît dans les images issues de techniques d'imagerie cohérente, comme l'holographie numérique, lorsque l'objet observé subit une déformation due à une sollicitation externe, qu'elle soit mécanique, acoustique, thermique ou pneumatique.

Cette déformation engendre un phénomène de décorrélation du speckle, se traduisant par l'apparition d'un bruit à composantes fréquentielles élevées, qui vient perturber le signal utile. [24]

II.7 Filtrage

C'est le cœur du processus : il s'agit de supprimer le bruit tout en conservant un maximum d'informations dans l'image d'origine. Selon la nature du bruit et les caractéristiques de l'image, différents algorithmes peuvent être utilisés. [19]

L'amélioration de la qualité des images repose principalement sur des techniques de filtrage qui poursuivent deux objectifs distincts : soit minimiser les bruits indésirables afin de restituer une image plus fidèle à la scène originale, soit accentuer les contours pour mieux faire ressortir les structures et détails significatifs de l'image.[9]

Le débruitage d'image est une étape essentielle en traitement d'images visant à éliminer les bruits tout en préservant les détails importants comme les contours et les textures.

II.8 Pourquoi appliquer un filtrage sur une image ?

Le filtrage d'une image est une opération essentielle en traitement d'image. Il permet d'améliorer la qualité visuelle de l'image ou d'en extraire des informations pertinentes pour une analyse ultérieure. Les objectifs principaux du filtrage sont les suivants :

Réduire ou éliminer le bruit : le filtrage permet de supprimer les perturbations (bruits) qui dégradent l'image, tout en préservant les détails importants.

Détecter les bords : certaines techniques de filtrage, comme les filtres de dérivation, sont utilisées pour faire ressortir les contours ou transitions dans l'image.

Effectuer une convolution : le filtrage consiste généralement à effectuer une convolution entre un noyau (filtre) et l'image, afin de transformer son contenu selon un but précis (lissage, renforcement, extraction...).

II.9 Les types des filtres

II.9.1 Filtre non adaptatif

II.9.1.1 Le filtrage moyennneur

Le filtre moyennneur est un filtre linéaire qui permet de réduire le bruit dans une image en remplaçant chaque pixel par la moyenne arithmétique des pixels situés dans son voisinage. Mathématiquement, il s'exprime comme une convolution entre l'image et un masque **H**, généralement une matrice $n \times n$ remplie de 1 et multipliée par $\frac{1}{n^2}$. [27]

L'expression du filtre moyennneur :

$$Valeur\ filtrée = \frac{1}{n^2} \sum_{i=1}^n \sum_{j=1}^n Pixel(i,j)$$

II.9.1.1.1 Forme du masque H

Dans la majorité des cas il s'agit d'une matrice 3×3 . Elle opère donc sur le pixel central via les huit pixels qui l'entoure et ceci s'effectue bien par le produit de convolution.

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

[27]

II.9.1.1.2 Exemple de fonctionnement

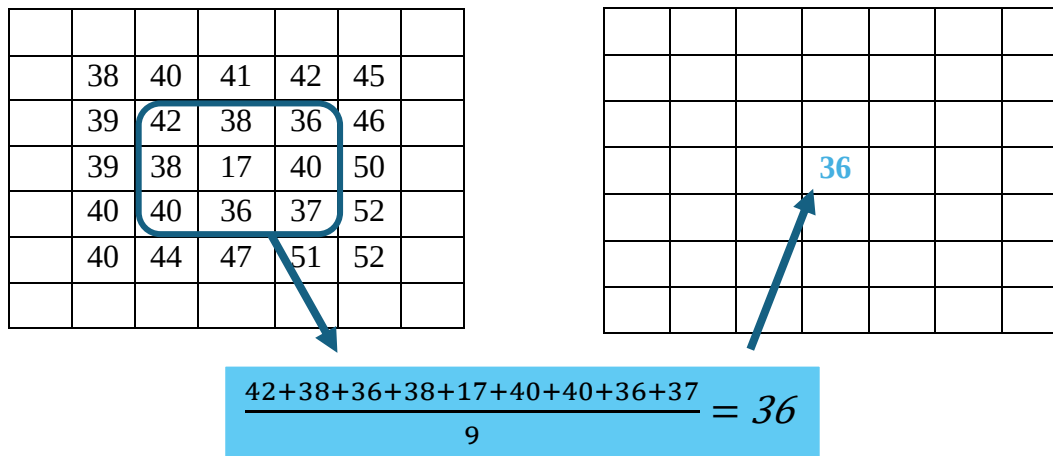


Figure II.1 : Illustration du fonctionnement du filtre moyenneur avec une fenêtre 3×3

II.9.1.2 Filtre gaussien

Le filtre gaussien est un filtre très utilisé en traitement d'image pour adoucir ou flouter une image. Il est basé sur la fonction gaussienne, une fonction en cloche qui donne plus de poids aux pixels proches du centre du noyau, réduisant ainsi le bruit tout en conservant les contours. [9]

II.9.1.2.1 Fonctionnement du filtre gaussien

Le filtre applique une convolution entre l'image et un noyau gaussien (aussi appelé masque ou kernel) défini par la fonction :

$$G(x, y) = \frac{1}{2\pi\sigma^2} \cdot e^{-\frac{x^2+y^2}{2\sigma^2}} \quad [10]$$

- **x, y** : les coordonnées par rapport au centre du noyau
- **σ** : l'écart-type, qui contrôle le niveau de flou (plus il est grand, plus le flou est prononcé)

Dans ce cas aussi, l'effet du filtre augmente avec la taille de son masque. Les contours et les détails fins sont cependant mieux conservés qu'avec le moyenneur. En effet, en utilisant une pondération gaussienne, le filtre gaussien ou smooth prend mieux en compte les corrélations entre pixels, notamment pour une texture d'image (la fonction de corrélation des niveaux de gris pour une texture est fréquemment modélisée par une gaussienne). Le filtre gaussien est un bon exemple pour les performances qu'on peut obtenir avec un filtre linéaire à réponse impulsionnelle finie. Le gros avantage de ces filtres, c'est leur facilité de conception et d'implémentation, mais ils ne peuvent être utilisés pour des travaux trop fins (la détérioration des contours qu'ils induisent par exemple, empêchera une segmentation fine des images). Voici, un exemple d'application du filtre Gaussien et du filtre Moyenneur sur une image en niveaux de gris :

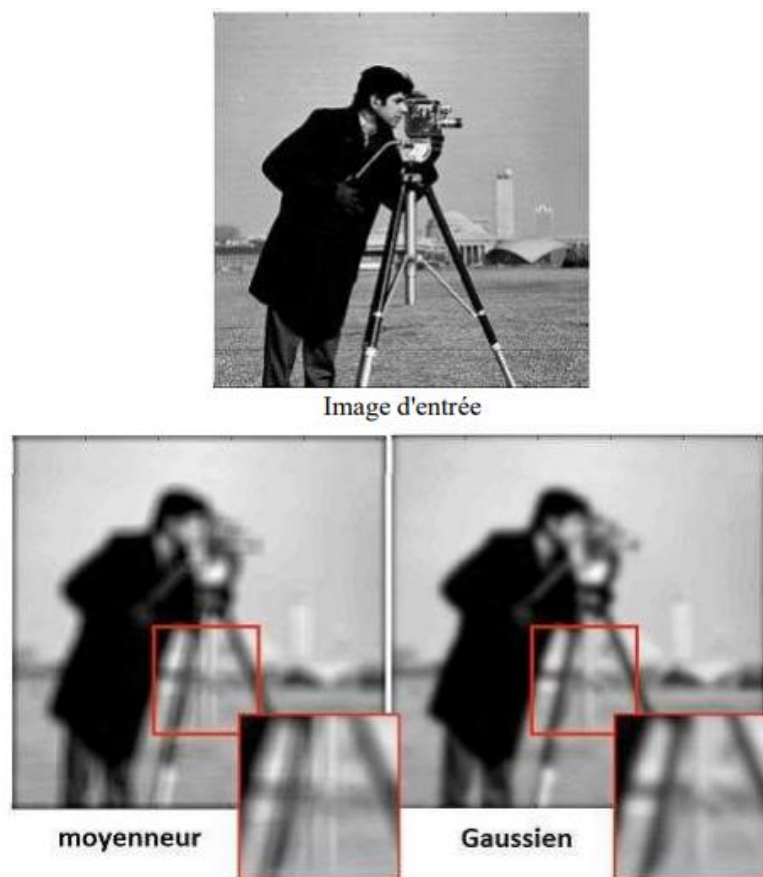


Figure II.2 : Exemple filtre moyenneur et gaussien.

II.9.1.2.2 Avantages du filtre gaussien

Meilleure préservation spectrale Contrairement au filtre moyenneur uniforme, le filtre gaussien ne provoque pas de recouvrement spectral car sa transformée de Fourier est elle-

même une gaussienne sans lobes secondaires. Cette propriété permet une transition plus douce dans le domaine fréquentiel, ce qui réduit les artefacts liés au filtrage.

Séparabilité Le filtre gaussien présente la propriété mathématique de séparabilité, ce qui permet d'optimiser considérablement les calculs. Concrètement :

- On peut d'abord filtrer les lignes avec un noyau 1D horizontal (1×5 par exemple)
- Puis filtrer le résultat (les colonnes) avec un noyau 1D vertical (5×1)

Cette séparabilité réduit significativement la complexité algorithmique :

- Pour un filtre 5×5 , une convolution 2D classique nécessite 25 multiplications et 24 additions, soit 49 opérations par pixel
- En exploitant la séparabilité, on ne requiert que 2×5 multiplications et 2×4 additions, soit seulement 18 opérations par pixel [66]

II.9.1.2.3 Inconvénient du filtre gaussien

Le principal désavantage du filtre gaussien réside dans l'utilisation de poids non unitaires pour les multiplications, ce qui augmente la complexité algorithmique comparativement au filtre moyennneur uniforme. Pour ce dernier, on peut se limiter aux additions dans l'opération de convolution, ce qui simplifie les calculs. [66]

II.9.1.3 Le filtrage médian simple

Le concept de médiane est une alternative robuste à la moyenne dans le domaine statistique. Pour un ensemble de n valeurs numériques $x_1, x_2, \dots, x_n$ (où n est impair), on détermine la médiane en ordonnant ces valeurs de façon croissante, puis en sélectionnant la valeur centrale située à la position $(n+1)/2$.

Contrairement au filtre moyennneur qui tend à flouter les contours des objets, le filtre médian présente l'avantage d'éliminer les parasites isolés dans l'image tout en préservant les contours. Ceci est dû à deux propriétés essentielles :

1. Un pixel aberrant dans le voisinage n'influence que très peu la valeur médiane.
2. Puisque la valeur médiane correspond toujours au niveau de gris d'un des pixels existants, aucun nouveau niveau de gris n'est introduit dans l'image.

Cette caractéristique permet au filtre médian de mieux conserver la netteté des contours marqués, particulièrement pour les éléments de taille supérieure au noyau du filtre. Cependant, il faut noter que les détails fins de l'image sont irrémédiablement supprimés par ce type. [2]

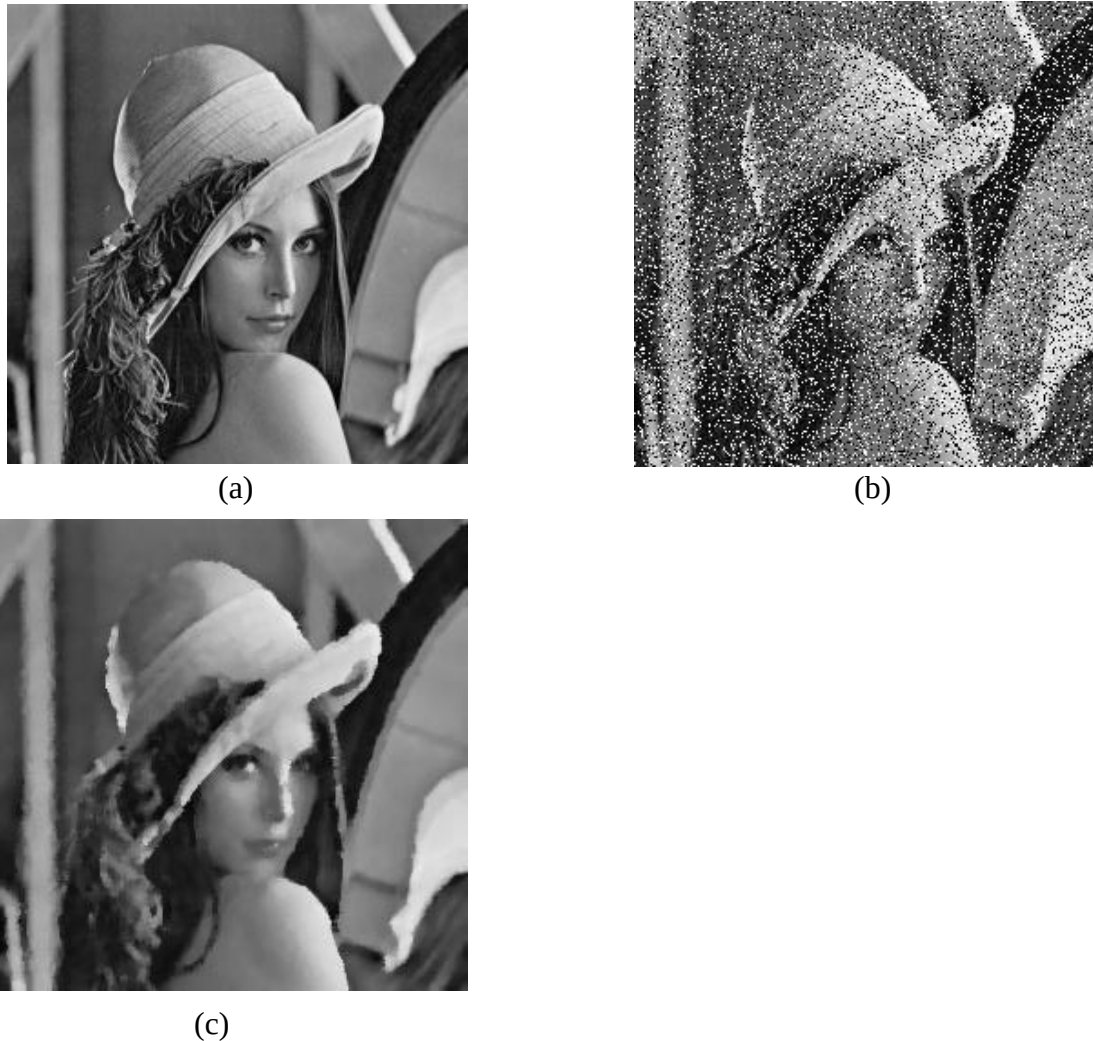
Exemple :

Figure II.3 : (a) image originale, (b) image avec bruit de sel et poivre élevé, (c) Filtrage médian 3×3 de l'image bruitée.

II.9.2 Filtre adaptatif

II.9.2.1 Filtre de Wiener

Le filtre de Wiener est un filtre linéaire et optimal qui minimise l'erreur quadratique moyenne entre le signal débruité et le signal original. [41]

II.9.2.1.1 Formulation mathématique du filtre de Wiener

II.9.2.1.1.1 Dans le domaine spatial

Pour une image bruitée $g(x,y)$, le filtre de Wiener produit une estimation du signal original $s(x,y)$ en utilisant :

$$\hat{s}(x, y) = \mu_g(x, y) + \frac{\sigma_g^2(x, y) - \sigma_n^2}{\sigma_g^2(x, y)} [g(x, y) - \mu_g(x, y)] \quad [2]$$

Où

- μ est la moyenne locale du signal
- σ_s^2 est la variance locale du signal
- σ_n^2 est la variance du bruit

Lorsque le filtre est appliqué en domaine spatial dans MATLAB, il est souvent implémenté à l'aide d'une fenêtre glissante autour de chaque pixel, où l'on estime la moyenne et la variance locales. Ainsi, dans les régions homogènes, le filtre réduit fortement le bruit, tandis que dans les zones à fort contraste, il préserve les contours.

II.9.2.1.1.2 Dans le domaine fréquentiel

L'expression du filtre de Wiener dans le domaine fréquentiel :

$$|G(f)| = \frac{|H(f)|S_x(f)}{|H(f)|^2S_x(f) + S_n(f)} \quad [13]$$

Où:

- $H(f)$ est la réponse en fréquence du système (filtre ou canal),
- $S_x(f)$ est la densité spectrale de puissance du signal d'entrée,
- $S_n(f)$ est la densité spectrale de puissance du bruit,
- $G(f)$ est la réponse fréquentielle optimale du filtre (filtre Wiener),

Ce comportement adaptatif rend le filtre de Wiener particulièrement utile dans le débruitage d'images affectées par un bruit gaussien blanc ou par des effets de flou modérés, tout en maintenant un bon compromis entre lissage et préservation des détails.

II.9.2.2 Filtre médian

Le filtre médian est un filtre non linéaire largement utilisé en traitement d'images numériques, principalement dans le cadre de la réduction de bruit. Cette opération constitue une étape de prétraitement essentielle, permettant d'améliorer la qualité de l'image et d'optimiser les performances d'algorithmes ultérieurs, tels que la détection de bords ou la segmentation. [20] [21]

Ce filtre est particulièrement efficace pour atténuer le bruit impulsionnel (bruit "sel et poivre") tout en préservant les contours et les détails importants de l'image, ce qui constitue un avantage majeur par rapport aux filtres linéaires classiques. [20] [21]

II.9.2.2.1 Principe de fonctionnement

1. **Parcours de l'image** : Pour chaque pixel de l'image, on considère un **voisinage** (une fenêtre carrée de taille $N \times N$, souvent 3×3 ou 5×5).
2. **Tri des valeurs** : Les valeurs des pixels dans ce voisinage sont triées par ordre croissant.
3. **Sélection de la médiane** : La valeur médiane (celle au milieu de la liste triée) est choisie pour remplacer la valeur du pixel central.
4. Répéter cette opération pour tous les pixels de l'image.

Le filtrage médian peut être formellement défini par l'équation suivante :

$$g(x,y) = \text{median}_{(i,j) \in R(x,y)} \{f(i,j)\}$$

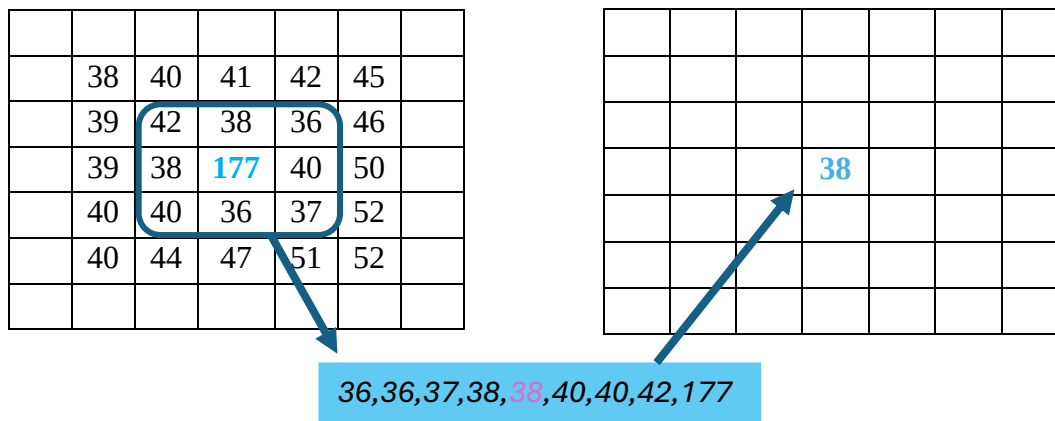


Figure II.4 : Illustration du fonctionnement du filtre médian avec une fenêtre 3×3

II.9.2.3 Filtre bilatéral

Le filtre bilatéral est un outil de lissage d'image préservant les bords bien connus. L'idée clé du filtre bilatéral consiste à introduire un poids photométrique dans le filtre gaussien standard. L'effet de ce poids est d'annuler les interactions spatiales entre les pixels présentant une différence d'intensité importante, préserve les contours tout en réduisant le bruit. Il a été introduit par C. Tomasi et R. Manduchi en 1998 s'inspire de principes utilisés dans la vision humaine. [7]

Contrairement au filtre gaussien qui ne tient compte que de la distance spatiale entre les pixels, le filtre bilatéral combine deux types de pondérations :

- Une pondération spatiale qui tient compte de la proximité des pixels dans l'image (comme dans le filtre gaussien);
- Une pondération radiométrique qui prend en compte la similarité d'intensité entre les pixels (pour préserver les bords).

II.9.2.3.1 Avantages

Préservation des contours : les pixels très différents en intensité ne sont pas pris en compte dans lissage.

Efficace contre le bruit gaussien et partiellement contre le bruit impulsionnel.

Applicable aux images couleur sans adaptation complexe. [7]

II.9.2.3.2 Inconvénients

Coût computationnel élevé, surtout pour de grandes fenêtres.

Sensible aux paramètres (σ_s , σ_r) qui doivent être choisis avec soin pour un bon compromis entre débruitage et préservation des détails. [7]

II.9.2.3.3 Utilisation

Le filtre bilatéral est largement utilisé dans :

- Le prétraitement des images médicales.
- La photographie numérique (effet de "lissage de peau").
- Les algorithmes de segmentation.
- Le traitement HDR et l'amélioration locale du contraste.

II.10 Comparaison pratique entre filtre moyenneur et médian

II.10.1 Avec bruit impulsionnel sel & poivre

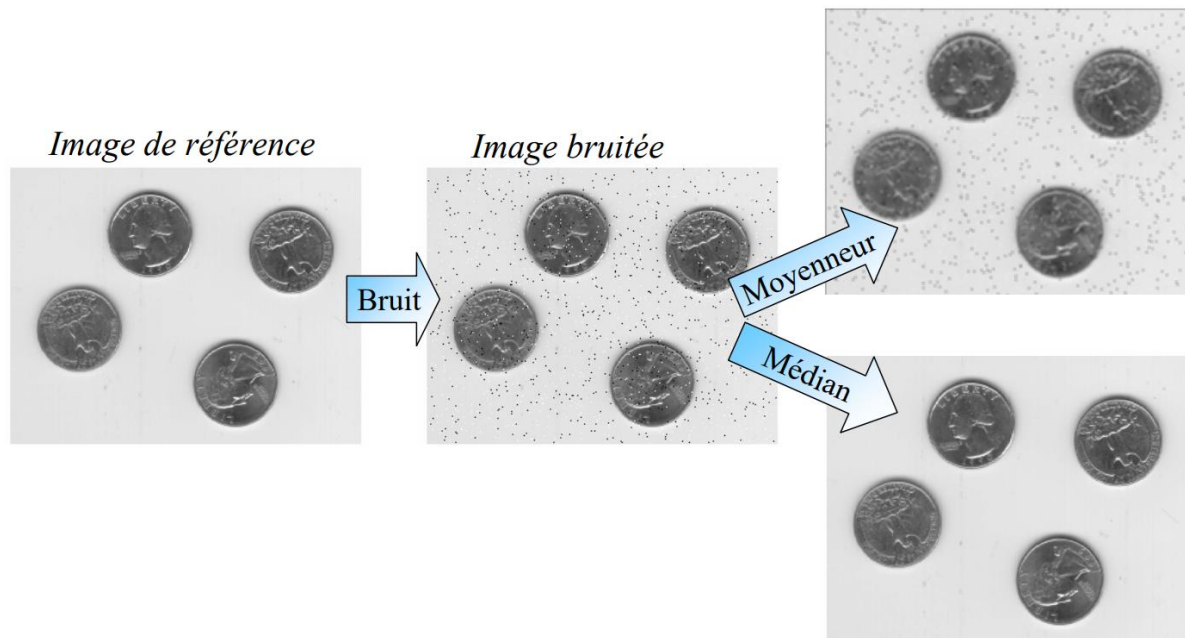


Figure II.5 : Comparaison des performances de débruitage de bruit sel et poivre entre filtre moyenneur et filtre médian.

II.10.1.1 Observation

Le filtrage médian est plus adapté que le filtrage moyenneur pour réduire le bruit impulsionnel.

II.10.2 Avec bruit gaussien

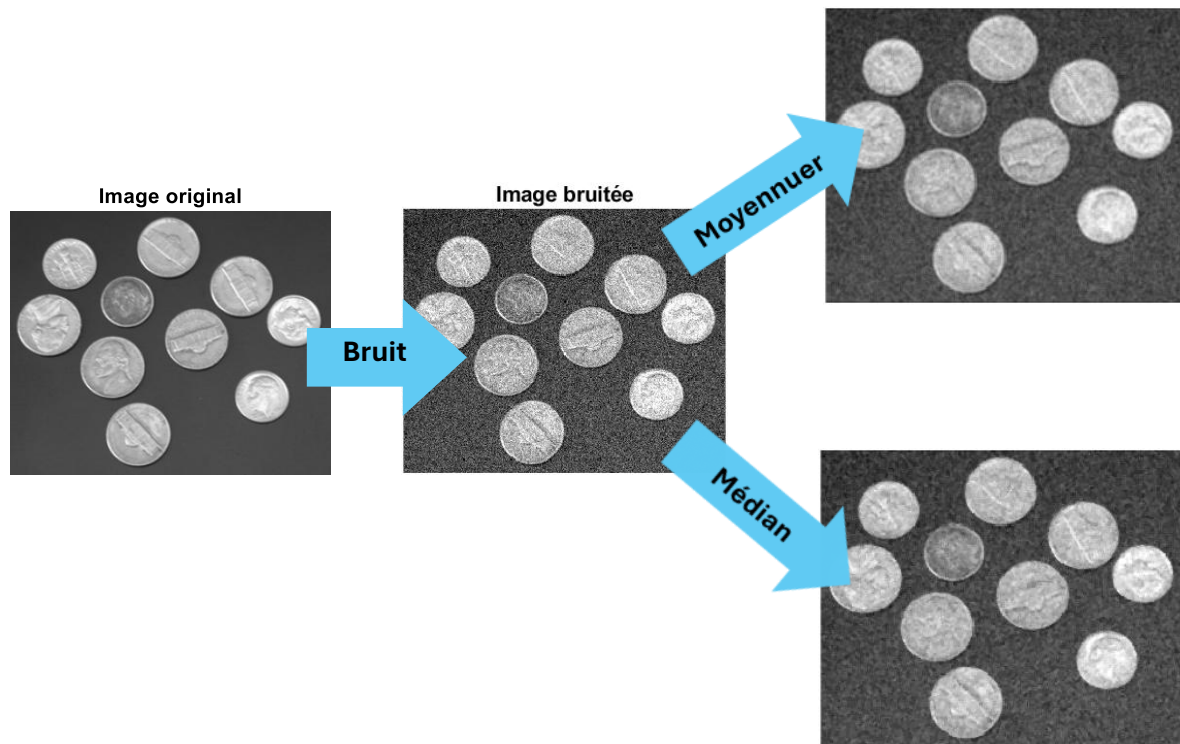


Figure II.6 : Comparaison des performances de débruitage de bruit gaussien entre filtre moyenneur et filtre médian

II.10.2.1 Observation

Pour le traitement du bruit gaussien, le filtre médian démontre une supériorité claire par rapport au filtre moyenneur. Il offre un compromis optimal entre débruitage efficace et préservation des caractéristiques importantes de l'image, particulièrement visible dans la conservation des contours nets des pièces de monnaie.

II.11 La dégradation des images

Le processus de dégradation est modélisé comme une fonction de dégradation qui ajoute un terme de bruit additif à une image d'entrée $f(x,y)$ pour produire une image dégradée $g(x,y)$ (voir la Figure II.7). Etant donné $g(x,y)$, la fonction de dégradation H et le bruit additif $\eta(x,y)$, l'objectif de la restauration est d'obtenir une estimation $\hat{f}(x,y)$ de l'image originale. Nous voulons que l'estimation soit aussi proche que possible de l'image d'entrée originale et, en général, plus nous connaissons H et η , plus $\hat{f}(x,y)$ se rapproche de $f(x,y)$. L'approche de restauration utilisée dans ce chapitre est basée sur différents types de filtres de restauration d'image. [2]

Il est montré que si H est un processus linéaire, invariant à la position, l'image dégradée est donnée dans le domaine spatial par :

$$g(x,y) = h(x,y) \star f(x,y) + \eta(x,y) \quad [2]$$

Où $h(x,y)$ est la représentation spatiale de la fonction de dégradation et le symbole $\star$ indique la convolution.

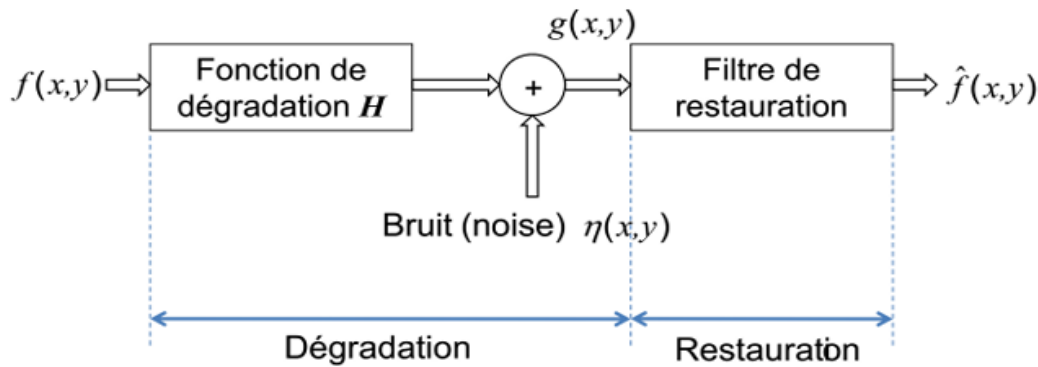


Figure II.7 : Un modèle pour le processus dégradation/restauration d'image.

Nous savons que la convolution dans le domaine spatial est analogue à la multiplication dans le domaine fréquentiel, nous pouvons donc écrire le modèle dans l'équation. Dans une représentation équivalente de domaine fréquentiel : (transfer de fourier).

$$G(u,v) = H(x,y)F(u,v) + N(u,v) \quad [2]$$

II.12 Conclusion

À travers ce chapitre, nous avons étudié les différents types de filtres utilisés en traitement d'images, en mettant l'accent sur les caractéristiques, avantages et limites de chaque méthode. Les filtres non adaptatifs comme le moyenneur ou le gaussien sont simples à appliquer, mais ont tendance à flouter les contours. À l'inverse, les filtres non linéaires tels que le filtre médian ou le filtre bilatéral permettent un meilleur compromis entre débruitage et préservation des détails.

Enfin, les filtres adaptatifs, comme le filtre de Wiener, s'ajustent aux propriétés locales de l'image, offrant des résultats plus performants dans des situations complexes. Cette diversité d'approches montre qu'il est important de choisir le filtre en fonction du type de bruit et de l'objectif visé dans chaque application.

Chapitre III:

Détection et

reconnaissance d'objet

III.1 Introduction

Dans le domaine de la vision par ordinateur, la capacité à comprendre le contenu d'une image repose sur plusieurs étapes essentielles, dont la segmentation, la classification et la reconnaissance d'objets. La segmentation constitue le point de départ de ce processus : elle consiste à diviser une image en régions homogènes, facilitant ainsi l'isolement des objets d'intérêt par rapport à leur fond ou aux autres éléments visuels.

Une fois cette étape réalisée, la classification permet d'attribuer à chaque région ou objet une étiquette ou une classe précise (par exemple : véhicule, personne, animal, etc.). Enfin, la reconnaissance vise à identifier avec précision la nature des objets, parfois jusqu'à leur modèle ou sous-catégorie.

Avec l'avènement de l'apprentissage profond, des approches performantes telles que les réseaux de neurones convolutifs (CNN), R-CNN, SSD et YOLO ont permis des avancées majeures dans ces domaines. Ce chapitre explore donc les fondements et les techniques modernes de la segmentation, de la classification et de la reconnaissance d'objets, en mettant en avant leur rôle complémentaire dans la compréhension automatique des images.

III.2 Introduction à la reconnaissance d'objets

La détection d'objets est une tâche essentielle en vision par ordinateur qui consiste à identifier et localiser automatiquement des objets dans une image ou une séquence vidéo. Contrairement à la simple classification d'image, elle permet de délimiter chaque objet à l'aide de boîtes englobantes et de les étiqueter par catégorie. Grâce à l'essor de l'apprentissage profond, notamment les réseaux neuronaux convolutifs (CNN), cette tâche a connu des avancées majeures. Des modèles comme YOLO, SSD ou Mask R-CNN offrent aujourd'hui des performances élevées et une détection en temps réel, ouvrant la voie à des applications concrètes telles que la conduite autonome, la vidéosurveillance intelligente, ou encore le contrôle de qualité industriel. La détection d'objets représente ainsi une passerelle entre la classification et la segmentation d'images, en combinant reconnaissance, localisation et catégorisation. [16]

III.3 Historique et évolution des techniques

La reconnaissance d'objets sur image a évolué depuis les années 1950, débutant par la détection de formes simples et la modélisation 3D à partir d'images 2D. Dans les années 1970-1990, des techniques classiques comme les opérateurs de contours Sobel et Canny ont

permis d'extraire des caractéristiques visuelles, tandis que les systèmes experts tentaient d'interpréter les images. Au début de la 21^{ème} siècle, l'apprentissage automatique a introduit des descripteurs locaux robustes (SIFT, SURF) et des classificateurs statistiques (SVM). La révolution du Deep Learning dans les années 2010 a transformé le domaine grâce aux réseaux de neurones convolutifs (CNN) capables d'apprendre directement à partir des données brutes, avec des architectures performantes telles que AlexNet et YOLO. Cette évolution a permis des progrès majeurs en précision et en rapidité, rendant la reconnaissance d'objets omniprésente dans de nombreux domaines industriels et scientifiques. [25] [26]

III.4 Détection vs Reconnaissance : Clarification des Concepts

Dans la vision par ordinateur, la détection d'objets consiste à localiser leur position dans une image via des boîtes englobantes, sans les identifier, tandis que la reconnaissance vise à identifier la nature ou la classe de chaque objet détecté. Ces deux tâches sont complémentaires : la détection répond à "où ?", la reconnaissance à "quoi ?". Leur combinaison est cruciale dans de nombreuses applications, comme la surveillance, la santé ou la conduite autonome, où il faut à la fois localiser et identifier les objets avec précision. [16]

III.5 Structure pour effectuer la classification des images

Pour réaliser une classification d'image, les étapes à suivre restent globalement les mêmes, quelle que soit la méthode employée :

Prétraitement de l'image

Le prétraitement d'une image vise à améliorer la qualité des données visuelles (caractéristiques) en corrigeant les distorsions et en renforçant les traits importants, afin de rendre les images optimales pour les modèles de vision par ordinateur. Il comprend plusieurs étapes : lecture de l'image, redimensionnement, réduction du bruit (par exemple grâce à un flou gaussien), normalisation ou étirement du contraste via la mise à l'échelle des niveaux de gris ou l'égalisation d'histogramme, ainsi que des techniques d'augmentation de données telles que la rotation, la translation ou la réflexion. Ces transformations permettent de garantir que l'image respecte les hypothèses des algorithmes et facilite une meilleure [56]

Détection d'objets : Cette étape consiste à localiser les objets présents dans l'image. Elle inclut la segmentation de l'image et l'identification des positions des objets d'intérêt.

Extraction et apprentissage des caractéristiques

L'extraction des caractéristiques constitue une phase essentielle dans le traitement d'image et l'apprentissage automatique. Elle vise à transformer les données brutes, telles que les pixels d'une image, en informations numériques plus simples à exploiter pour un modèle. Ces informations, appelées caractéristiques ou attributs, permettent de décrire certains aspects de l'image comme les formes, les contours, les couleurs ou les textures. Dans les méthodes classiques, ces caractéristiques sont choisies manuellement selon des critères visuels précis. En revanche, les méthodes modernes, comme les réseaux neuronaux profonds, permettent au modèle d'apprendre automatiquement ces caractéristiques à partir des données. Cette étape permet ainsi de distinguer les objets ou classes présents dans une image et de rendre possible leur classification ou leur reconnaissance. Elle joue un rôle fondamental dans la performance globale d'un système de vision par ordinateur. [57]

Classification des objets : Enfin, les objets détectés sont classés dans des catégories prédéfinies à l'aide d'un algorithme de classification approprié, qui compare les caractéristiques extraites aux modèles de référence pour attribuer une étiquette à chaque objet. [1]

III.6 Segmentation

La segmentation d'image est une étape essentielle du traitement d'image visant à regrouper les pixels en régions homogènes selon des critères définis à l'avance, tels que l'intensité (niveaux de gris), la texture, ou d'autres descripteurs (moments, ondelettes, etc.). Cette opération divise l'image en sous-ensembles cohérents appelés régions, formant une partition complète de l'image, c'est-à-dire que leur union reconstitue l'image originale. Un cas particulier de segmentation, lorsque l'on distingue uniquement deux classes (objet/fond), est appelé binarisation. La segmentation n'est généralement pas une finalité en soi, mais constitue une étape préalable adaptée au type de traitement ou d'analyse ultérieure envisagé (reconnaissance d'objets, détection de contours, etc.). [42][60]

III.6.1 Définition et objectifs

Techniquement, la segmentation est une partition de l'ensemble des pixels de l'image en groupes (régions), chaque groupe désignant un « objet » ou une zone distinctive. Cette étape intervient souvent après le prétraitement et avant l'extraction de caractéristiques ou la reconnaissance. [31]

Ses objectifs principaux sont :

- Extraire des régions homogènes selon un critère donné (intensité, couleur, texture).
- Réduire le bruit et isoler des structures pertinentes.
- Localiser précisément les contours ou les limites entre régions. [32]

III.6.2 Principales approches de segmentation

III.6.2.1 Segmentation par contours

L'approche par contours repose sur l'hypothèse que les frontières des objets dans une image correspondent à des changements brusques d'intensité lumineuse. Un contour est ainsi défini comme un ensemble de points où l'intensité varie fortement, marquant généralement la séparation entre deux régions homogènes, soit en termes de niveaux de gris, soit en termes de texture. [33]

Cette méthode de segmentation consiste donc à détecter les transitions significatives dans l'image afin de localiser les frontières entre les différentes zones visuelles. Elle se base sur l'extraction de lignes de contraste pour délimiter les objets présents.

Les techniques de segmentation par contours peuvent être regroupées en trois grandes catégories :

Les méthodes dérivatives : qui utilisent les gradients et les opérateurs de dérivation comme Sobel pour repérer les zones de variation rapide. [34]

Les méthodes analytiques : qui s'appuient sur des modèles mathématiques et des algorithmes de détection comme Canny. [33]

Les méthodes déformables : comme les courbes actives (snakes) ou les niveaux de contours (level sets), qui adaptent dynamiquement les contours à la forme des objets dans l'image. [31]
[33]

III.6.2.2 Segmentation par régions

Les approches par régions, considérées comme duales des méthodes basées sur les contours, visent à identifier les zones homogènes au sein d'une image. Elles consistent à regrouper les pixels partageant des caractéristiques communes selon des critères locaux, tels que le niveau de gris, la moyenne, la variance, ou encore des descripteurs de texture calculés dans le voisinage du pixel. Ces régions homogènes sont ensuite différenciées les unes des autres par ces mêmes attributs. On trouve par exemple la croissance de régions, la division/fusion ou les méthodes basées sur la classification non supervisée (comme k-means). [35]

III.6.2.3 Segmentation hybride

La segmentation hybride combine plusieurs méthodes de segmentation (généralement des approches basées sur les régions et les contours) afin de compenser les limites de chacune et d'obtenir des résultats plus robustes et précis. Par exemple, dans un cadre médical, on peut commencer par une segmentation régionale (fuzzy connectedness) pour identifier les zones homogènes. Une approche hybride typique intègre donc des étapes complémentaires (détection initiale de régions, puis raffinement des frontières), permettant une automatisation accrue et une segmentation plus fidèle aux structures réelles de l'image. [45]

III.6.2.4 Segmentation basée sur l'apprentissage

La segmentation basée sur l'apprentissage est une méthode avancée de traitement d'images qui utilise des techniques d'intelligence artificielle, principalement des réseaux de neurones profonds (deep learning), pour diviser une image en régions ou segments significatifs. Contrairement aux méthodes classiques qui reposent sur des règles ou des filtres basés sur les caractéristiques visuelles simples (couleur, intensité), cette approche apprend automatiquement à détecter et à classifier chaque pixel en fonction de modèles complexes appris à partir de grandes bases de données annotées. Plus précisément, la segmentation par apprentissage profond réalise une classification au niveau du pixel, ce qui permet d'identifier précisément les objets ou les zones présentes dans une image, même dans des contextes visuels complexes. Des architectures célèbres comme U-Net, Mask R-CNN ou DeepLab sont notamment utilisées pour ce type de tâches, avec d'excellents résultats notamment en imagerie médicale, vision industrielle ou conduite autonome. [46]

III.6.3 Techniques et méthodes

III.6.3.1 Clustering (k-means, fuzzy c-means)

Le clustering est une technique de classification non supervisée visant à regrouper automatiquement des données similaires en clusters, sans connaissance préalable des classes. Parmi les méthodes les plus utilisées figurent le K-means et le Fuzzy C-means (FCM). Le K-means affecte chaque donnée à un seul groupe, en fonction de la proximité avec le centre du cluster le plus proche, tandis que le FCM introduit une approche plus souple, en attribuant à chaque donnée un degré d'appartenance à plusieurs clusters, ce qui permet de mieux gérer les frontières floues entre les régions. Ces deux méthodes sont largement utilisées dans la segmentation d'images médicales, notamment pour détecter des structures complexes comme les vaisseaux rétiniens. Bien que le FCM offre souvent une précision supérieure, il est plus

coûteux en termes de calculs, contrairement au K-means, qui est plus rapide mais moins souple en cas de chevauchement entre classes. [47]

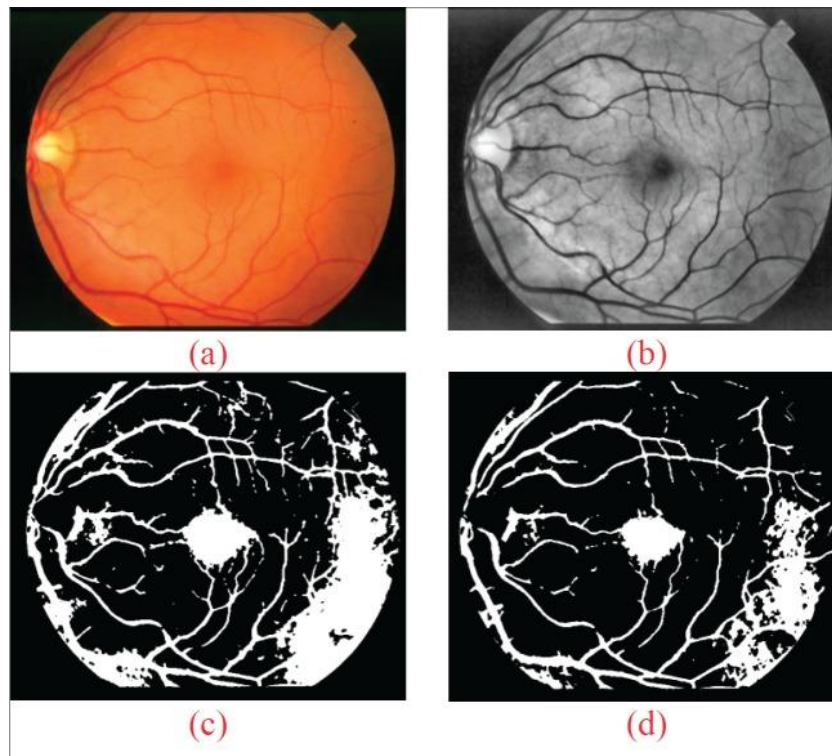


Figure III.1 : présente un nombre de clusters de 10 et la méthode de détermination du seuil avec une médiane. (a) présente un exemple d'image rétinienne du système d'entrée, (b) présente la sortie du processus CLAHE et du filtre médian. (c) présente une sortie combinée des k-moyennes et du seuillage, (d) présente la FCM et le seuillage.

III.6.3.1.1 L'algorithme K-moyens (ou K-means)

Le K-means est un algorithme simple et populaire utilisé pour classer des données (comme les pixels d'une image) en k groupes appelés clusters. Chaque élément est regroupé avec ceux qui lui ressemblent le plus, en se basant sur une distance, souvent la distance euclidienne.

Le principe est de trouver les centres de chaque groupe (appelés centroïdes) et affecter les points les plus proches. [36]

L'algorithme commence par choisir k centres au hasard, puis répète deux étapes :

1. Regrouper les éléments autour des centres les plus proches.
2. Recalculer les centres à partir des éléments affectés.

Ce processus continue jusqu'à ce que les centres ne bougent presque plus. En traitement d'image, K-means est souvent utilisé pour segmenter ou simplifier les images, en regroupant les zones similaires selon leur intensité ou couleur. C'est une méthode rapide, efficace et facile à mettre en œuvre. [36]

Avantages :

- 1) Le K-means est une méthode non supervisée, ne nécessitant aucune connaissance préalable des données, ce qui la rend utile pour découvrir des structures cachées.
- 2) Elle est applicable à divers types de données, à condition de choisir une mesure de distance adaptée. [44]

Inconvénients :

- 1) La performance dépend fortement du choix de la distance, ce qui peut être complexe si les données sont hétérogènes.
- 2) La méthode est sensible aux paramètres, notamment au nombre de clusters k , dont une mauvaise estimation peut dégrader les résultats.
- 3) Les groupes formés sont parfois difficiles à interpréter, rendant leur signification peu évidente à analyser. [44]

Segmentation par k-means

Figure III.2 : Résultat de la segmentation d'une image cameraman par la méthode k-means

III.6.3.1.2 L'algorithme de Fuzzy C-Means (FCM)

Le Fuzzy C-Means est un algorithme de clustering flou (ou « regroupement flou ») utilisé dans l'apprentissage non supervisé. Contrairement à K-Means, où chaque point appartient à un seul cluster, FCM permet à chaque point de données d'appartenir à plusieurs clusters simultanément, avec un degré d'appartenance (membership) compris entre 0 et 1. [11]

Il s'agit d'une méthode itérative qui :

- Initialise des centres de clusters aléatoires.
- Calcule les degrés d'appartenance de chaque point à chaque cluster.

- Met à jour les centres de clusters en fonction de ces degrés.
- Répète jusqu'à convergence.

Avantages de FCM

1. **Plus flexible que K-Means** : permet des appartenances partielles, ce qui est utile quand les frontières entre groupes ne sont pas nettes.
2. **Résistance au bruit** : les données bruitées ou ambiguës ont une influence réduite, car elles ne sont pas entièrement affectées à un seul cluster.
3. **Meilleure interprétation** : le degré d'appartenance offre une compréhension plus fine des relations entre les données. [11]

Inconvénient de FCM

1. **Coût computationnel élevé** : il faut calculer les degrés d'appartenance pour chaque point par rapport à chaque cluster à chaque itération.
2. **Paramètres à régler** : choisir le bon nombre de clusters et le paramètre de flou peut être difficile sans connaissance préalable.
3. **Sensibilité aux valeurs initiales** : comme K-Means, FCM peut converger vers des solutions locales si l'initialisation est mauvaise.
4. Évidente à analyser. [11]

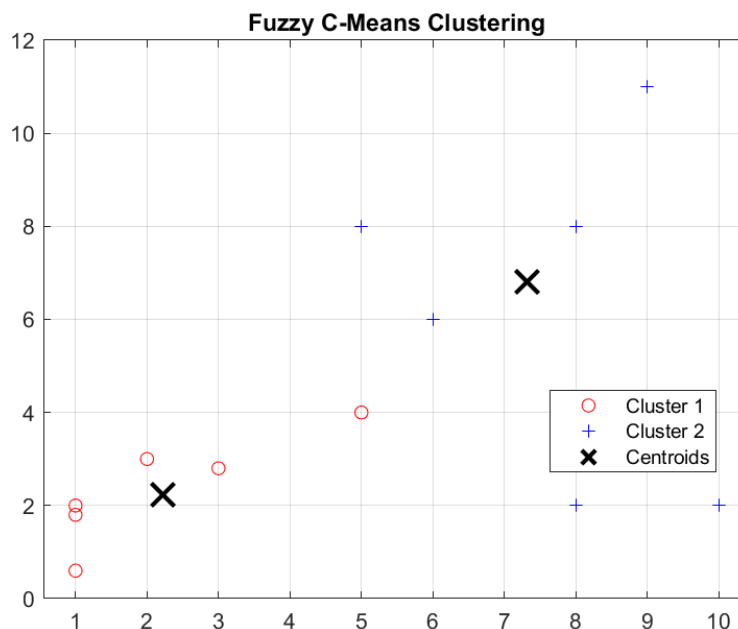


Figure III.3 : Résultat du regroupement Fuzzy C-means sur un jeu de données 2D.

Application sur image

Image segmentée avec FCM



Figure III.4 : Résultat de la segmentation d'une image cameraman par la méthode FCM

III.6.3.2 Seuillage

Le seuillage est une technique fondamentale de segmentation qui consiste à transformer une image en niveaux de gris en une image binaire. Concrètement, on fixe une valeur seuil, et pour chaque pixel de l'image :

- Si la valeur du pixel est supérieure ou égale au seuil, le pixel est assigné à la valeur maximale (par exemple 255, ce qui correspond au blanc).
- Sinon, il est assigné à 0 (noir).

Cette transformation permet de séparer les objets d'intérêt du fond, en simplifiant l'image pour faciliter son analyse ou son traitement ultérieur. [48]

Mathématiquement, si $f(i,j)$ est la valeur d'intensité du pixel en position (i,j) dans l'image originale, alors l'image seuillée $g(i,j)$ est définie par :

$$g(i,j) = \begin{cases} 255 & \text{si } f(i,j) \geq s \\ 0 & \text{sinon} \end{cases}$$

Où s est le seuil choisi. [49]

III.6.3.2.1 Méthode Otsu

La méthode d'Otsu, développée par Nobuyuki Otsu en 1979, est une technique de seuillage automatique global visant à séparer les pixels d'une image en deux classes – premier plan (objet) et arrière-plan – sans paramètre externe. Elle repose sur le calcul du seuil optimal qui minimise la variance intra-classe, équivalent à maximiser la variance inter-classe, tirée de l'histogramme des niveaux de gris de l'image. [65]



Figure III.5 : Illustration de la méthode d'Otsu. (a) Image originale en niveaux de gris. (b) Résultat du seuillage automatique par la méthode d'Otsu avec un seuil calculé de 0,42533.

III.6.3.3 Méthodes morphologiques

Les transformations morphologiques sont des opérations de traitement d'image basées sur l'application d'un élément structurant que l'on superpose à chaque pixel de l'image. Cet élément, généralement de forme simple (étoile, carré, disque...), agit comme un tampon utilisé pour modifier localement la structure de l'image. Ces transformations peuvent être considérées à la fois comme des opérations géométriques et comme des filtres. Les quatre opérations fondamentales en morphologie sont : la dilatation, l'érosion, l'ouverture et la fermeture, [51] principalement appliquées aux images en niveaux de gris. L'ouverture et la fermeture étant des combinaisons respectives de l'érosion et de la dilatation, une compréhension approfondie de ces deux dernières est essentielle pour maîtriser les principes de la morphologie mathématique. Ces ensembles appartiennent généralement à l'espace discret bidimensionnel Z^2 , où chaque élément est un vecteur de coordonnées (x,y) , représentant la position d'un pixel (blanc ou noir, selon la convention adoptée) dans l'image. Cette formalisation permet d'appliquer des opérations géométriques structurées pour analyser et transformer les images. [52]

III.6.3.3.1 Dilatation

La dilatation permet d'éliminer les points noirs [53]. par l'équation mathématique suivant :

$$A \oplus B = \{z | (\hat{B})_z \cap A \neq \emptyset\}$$

Soient deux ensembles A et B, la dilatation de A par B, notée $A \oplus B$, est définie comme l'ensemble des positions z pour lesquelles l'élément structurant B, translaté en z, intersecte l'ensemble A de manière non vide. En d'autres termes, la dilatation correspond à tous les décalages possibles de B pour lesquels au moins un pixel de B recouvre un pixel appartenant à A. Cette opération permet ainsi d'agrandir les objets présents dans une image binaire, en étendant leurs contours selon la forme de l'élément structurant. [52]

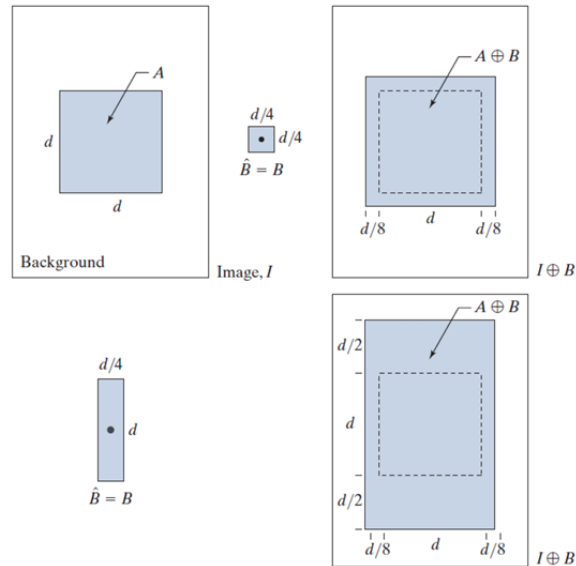


Figure III.6 : Illustration de l'effet de la dilatation sur une forme carrée selon différents éléments structuraux [52]

III.6.3.3.2 L'érosion

L'érosion permet d'éliminer les points blancs par l'équation mathématique suivant :

$$A \ominus B = \{z | (B)_z \subseteq A\} \quad [52]$$

L'érosion est définie de manière analogue à la dilatation, à la différence qu'elle repose sur l'opérateur MIN au lieu de MAX, ce qui revient à réduire les objets dans l'image en supprimant les pixels situés sur les contours, selon la forme de l'élément structurant utilisé.[54]

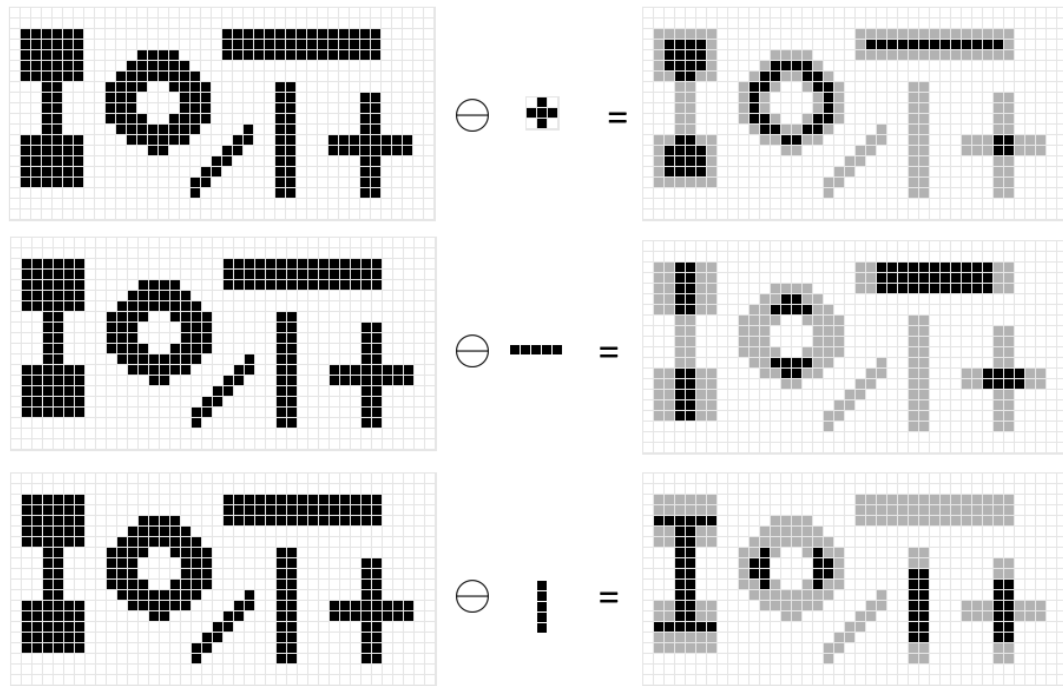


Figure III.7 : Illustration des effets de l'érosion selon différents éléments structuraux [54]

III.6.3.3 Ouverture & Fermeture

L'**ouverture** est une opération morphologique composée d'une érosion suivie d'une dilatation de l'image par un élément structurant.

Notée : $X \circ B = (X \ominus B) \oplus B$.

Elle permet d'éliminer les petits objets ou détails fins tout en conservant les formes principales. Elle adoucit les contours, bouche les petits trous et rend plus distinctes certaines régions comme les cheveux ou les doigts dans l'image observée.

La **fermeture** à l'inverse, consiste en une dilatation suivie d'une érosion.

Notée : $X \cdot B = (X \oplus B) \ominus B$.

Elle est utilisée pour combler les petits trous et supprimer les petites tâches tout en préservant la forme globale des objets. Elle est efficace pour unifier des zones fragmentées, comme le visage dans l'exemple donné, où les yeux (trous) sont comblés. [64]

Ces deux opérations sont fondamentales pour préparer les images à des étapes ultérieures de traitement (comme la segmentation), en nettoyant le bruit morphologique tout en conservant l'essentiel des structures.

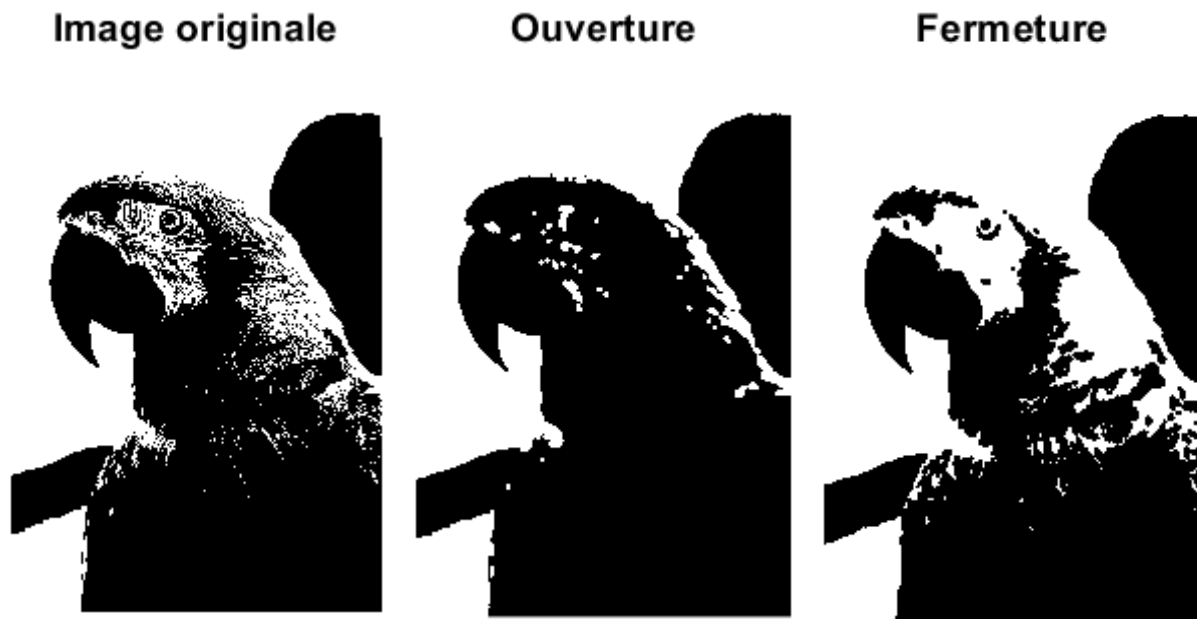


Figure III.8 : Effets des opérations morphologiques d'ouverture et fermeture sur une image binaire.

III.6.4 Domaines d'application principaux de la segmentation d'images

La segmentation trouve des applications dans de multiples domaines :

- 1) **Reconnaissance d'objets** : distinguer et isoler différents objets dans une image.
- 2) **Imagerie médicale et diagnostique assistée par ordinateur** : utilisée pour l'analyse de l'imagerie médicale, les scanners et les IRM.
- 3) **Vision par ordinateur et traitement d'images** : utilisée pour les contours, les textures ou les zones d'intérêt.
- 4) **Cartographie et télédétection** : analyser les zones géographiques et détecter des changements environnementaux comme la déforestation ou l'urbanisation.
- 5) **Industrie et robotique** : La segmentation d'images permet aux robots d'identifier et localiser des objets pour automatiser des tâches comme l'assemblage ou le contrôle qualité en industrie. [55]

III.7 Type de classification

III.7.1 Classification supervisée de l'image

La classification supervisée des pixels, également appelée classification avec apprentissage, consiste à construire une fonction de discrimination qui permet de diviser l'espace de représentation des données en se basant sur une connaissance a priori de l'image. Elle repose sur la création d'une base d'apprentissage, constituée à partir d'une segmentation de

référence, souvent réalisée manuellement par un ou plusieurs experts. Ces derniers procèdent à un étiquetage des régions d'intérêt dans l'image, en les répartissant selon C classes définies (notées $C_1, C_2, \dots, C_C$). Chaque classe C_i se voit ainsi associée à un ensemble d'apprentissage E_i , qui servira de référence pour entraîner le classificateur. Les données de test sont ensuite automatiquement segmentées en fonction de ces ensembles E_i . Le choix rigoureux de la taille et de la représentativité de la base d'apprentissage est fondamental, car il influence directement la fiabilité et la précision de la classification supervisée appliquée à l'image. [32][50]

III.7.2 Classification non supervisée de l'image

Pour effectuer une classification des données, il est d'abord nécessaire d'en extraire les caractéristiques pertinentes, qui seront ensuite exploitées par un algorithme de classification. La classification non supervisée, également appelée classification sans apprentissage, vise à regrouper des pixels ou des individus en partitions homogènes, en se basant sur la proximité de leurs vecteurs d'attributs dans un espace de représentation, et ce, sans connaissance préalable sur les classes existantes. Cette approche permet ainsi d'effectuer une segmentation automatique, dite "en aveugle", particulièrement utile dans le traitement d'images. Dans ce cadre, nous nous intéressons à deux méthodes couramment utilisées pour la classification non supervisée des pixels :

- l'algorithme des K-moyennes (K-means).
- l'algorithme des C-moyennes floues (Fuzzy C-means). [43]

III.8 détection et reconnaissance d'objet

Il existe deux grandes catégories de détecteurs d'objets. La première regroupe les détecteurs en deux étapes, tels que Faster R-CNN ou Mask R-CNN. Ces modèles commencent par générer des régions d'intérêt à l'aide d'un réseau de propositions, puis soumettent ces régions à un second module chargé de classifier les objets et d'ajuster les boîtes englobantes.

La seconde catégorie concerne les détecteurs en une seule étape, comme YOLO et SSD, qui abordent la détection d'objets comme un problème de régression directe. Ils prennent une image en entrée et prédisent simultanément les probabilités de classe ainsi que les coordonnées des boîtes de délimitation. Bien que leur précision soit généralement inférieure à celle des détecteurs en deux étapes, ils sont nettement plus rapides.

Il existe de nombreuses méthodes utilisées dans le domaine de la détection humaine-objet.

III.8.1 R-CNN (Region-based Convolutional Neural Network)

Le modèle de détection d'objets R-CNN (Regions with CNN features) a été introduit par Ross Girshick en 2014 avec une architecture de réseau profond destinée à la détection d'objets dans des images. Elle vise à localiser et classer plusieurs objets au sein d'une image en produisant des boîtes englobantes (bounding boxes) associées à une classe [58]. L'algorithme suit un processus en trois étapes principales :

Génération de propositions régionales comme Selective Search pour générer environ 2 000 régions candidates (region proposals) susceptibles de contenir un objet.

Extraction de caractéristiques : chaque région proposée est ensuite traitée par un réseau neuronal convolutionnel profond, qui en extrait un vecteur de caractéristiques de taille fixe.

Classification et localisation : enfin, un ensemble de classificateurs SVM linéaires, spécifiques à chaque classe, permet de classer les objets, tandis qu'un régresseur affine les prédictions de localisation.

En résumé, R-CNN combine la recherche de régions d'intérêt avec l'extraction de caractéristiques CNN, puis la classification et la localisation des objets pour obtenir une segmentation grossière par boîtes prédictives. [59]

Avantages de R-CNN

- **Précision élevée** : Fournit une détection d'objets très précise grâce à l'utilisation de caractéristiques extraites localement par CNN.
- **Robuste aux variations d'objets** : Gère efficacement les objets de différentes tailles, formes, orientations et contextes.
- **Flexible** : Peut être adapté à diverses tâches comme la segmentation d'instances ou le suivi d'objets, en modifiant simplement la sortie du réseau. [59]

Inconvénients de R-CNN

- **Complexité computationnelle** : Le processus en plusieurs étapes (proposition de régions, CNN, classification) est lourd et coûteux en ressources.
- **Lenteur en inférence** : Traite chaque région séparément, ce qui ralentit fortement la détection, rendant le modèle inadapté aux applications temps réel.
- **Redondance des régions** : Génère souvent des propositions qui se chevauchent, entraînant des calculs inutiles.
- **Modèle non end-to-end** : Ne permet pas une optimisation conjointe des étapes, contrairement à des architectures plus modernes comme Faster R-CNN. [59]

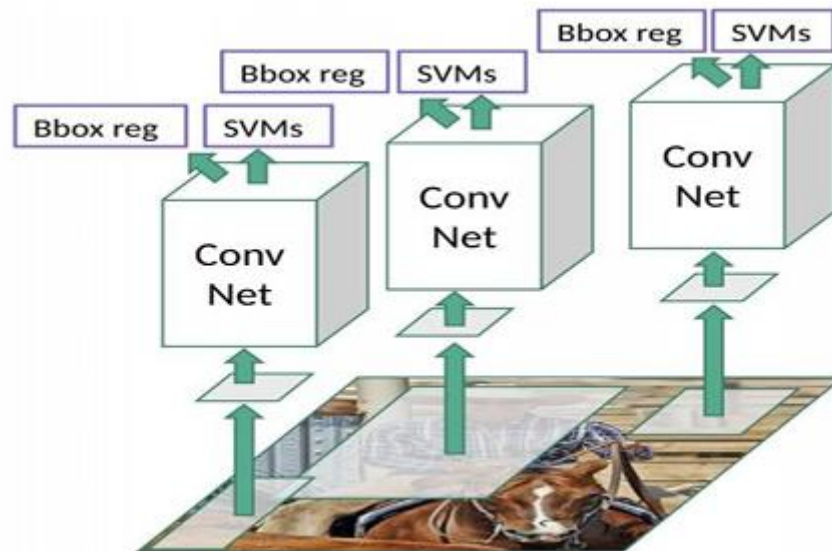


Figure III.9: Architecture R-CNN

III.8.3 Faster R-CNN

Faster R-CNN (Regions with Convolutional Neural Networks) est une méthode avancée de détection d'objets basée sur l'apprentissage profond.

Elle combine un réseau convolutionnel pour l'extraction de caractéristiques et un RPN (Region Proposal Network) pour générer des propositions de régions.

Ces propositions sont ensuite affinées et classifiées pour localiser les objets avec précision. Faster R-CNN est structuré en deux parties : la génération de régions et la détection d'objets. Il est entièrement entraînable de bout en bout et offre un bon compromis entre précision et vitesse.

Grâce au partage des couches de convolution, il est plus rapide que les modèles précédents. Cette approche est adaptée aux applications exigeant une haute précision de détection. Elle est largement utilisée en surveillance, reconnaissance d'objets et analyse d'images. [67]

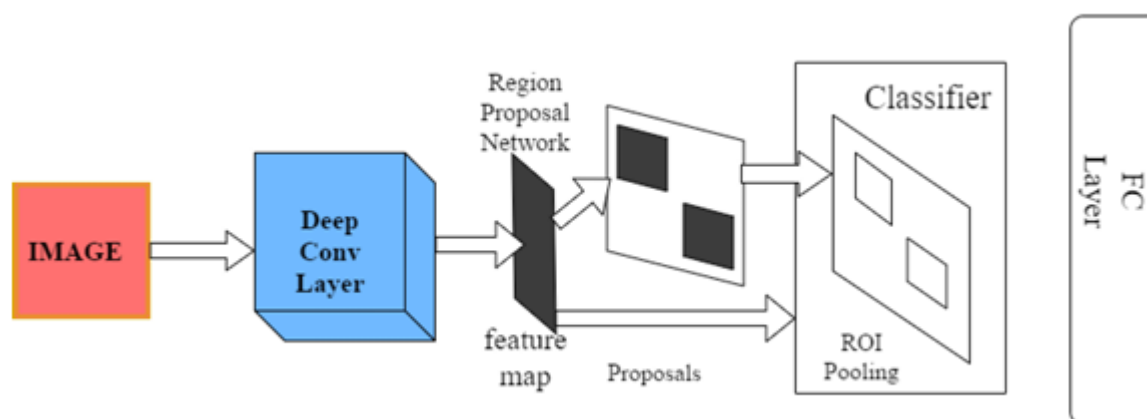


Figure III.10: Architecture de model Faster R-CNN

III.8.4 Mask R-CNN

Mask R-CNN (Mask Region-based Convolutional Neural Network) est un modèle de détection d'objets et de segmentation d'instances construit à partir de Faster R-CNN. Il ajoute une branche dédiée à la prédiction de masques (mask head) en parallèle avec les sorties de classification et de localisation. Grâce à la technique de RoIAlign, il corrige les erreurs d'alignement qui limitaient la précision des versions précédentes. Le modèle peut ainsi générer des masques binaires précis à l'échelle pixel pour chaque objet détecté, tout en maintenant une architecture efficace et entraînable de bout en bout. [70]

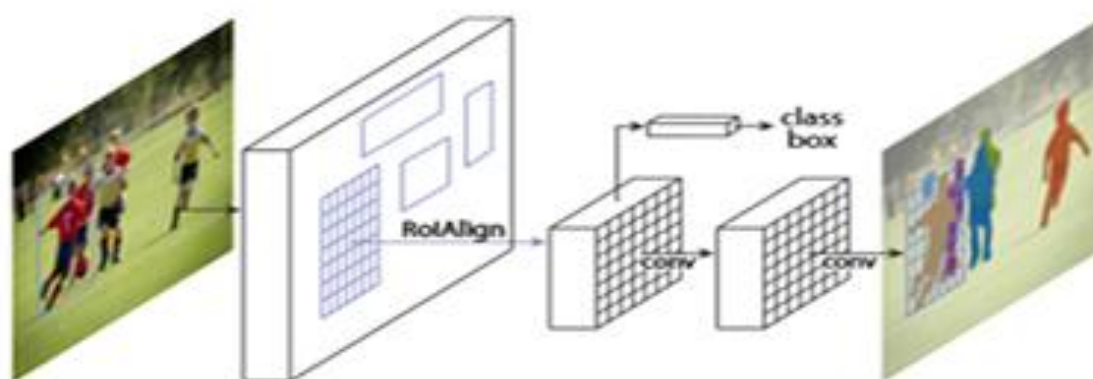


Figure III.11 : Architecture de Mask R-CNN avec la couche RoIAlign

III.8.5 Single Shot MultiBox Detector

SSD (Single Shot MultiBox Detector) est une architecture de détection d'objets en une seule passe (one-shot), qui supprime le besoin de générer des propositions de régions séparément. Elle découpe l'espace des boîtes englobantes possibles en un ensemble de boîtes par défaut ("default boxes") réparties sur plusieurs cartes de caractéristiques à différentes échelles et

ratios. Pour chaque boîte, le réseau prédit à la fois la probabilité de présence d'un objet et les ajustements géométriques nécessaires à la boîte pour qu'elle corresponde précisément à l'objet détecté. En combinant des prédictions issues de plusieurs couches de la CNN, SSD traite efficacement les objets de tailles variées. Cette approche permet d'obtenir une détection rapide (temps réel) tout en conservant une grande précision, et offre une solution d'apprentissage unifiée (end-to-end) applicable à des tâches exigeantes comme la surveillance ou la conduite autonome. Les premières couches du réseau s'appuient sur une architecture standard performante pour la classification d'images, à savoir VGG-16, tronquée avant les couches de classification. Cette partie constitue le réseau de base. [61]

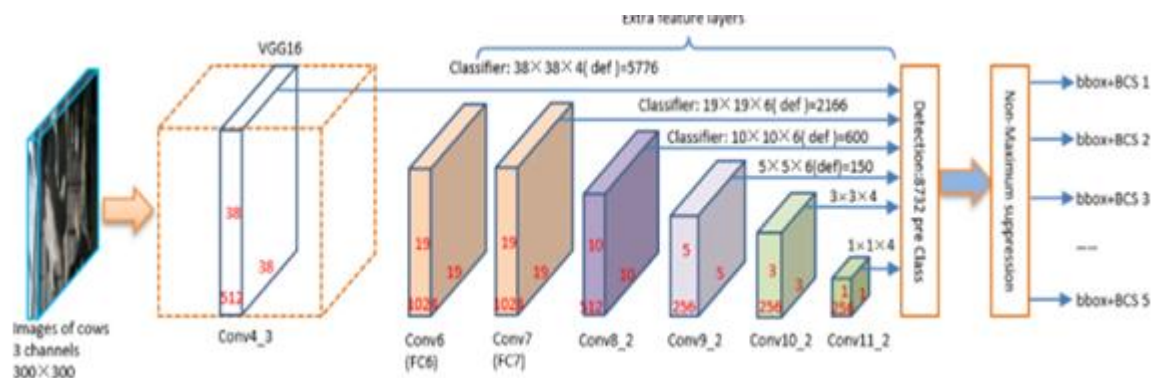


Figure III.12 : architecture de model SSD

Le SSD est un détecteur d'objets en une seule étape qui combine des prédictions à partir de caractéristiques multi-échelles. Il est plus rapide que les détecteurs en deux étapes comme Faster R-CNN, tout en étant plus précis que les détecteurs à une seule échelle comme YOLOv2. [62]

III.8.6 YOLO : You Only Look Once

YOLO, introduit par Joseph Redmon en 2015, est un algorithme de détection d'objets en temps réel. Ses versions améliorées, YOLOv2 (YOLO9000) et YOLOv3, ont respectivement été publiées en 2016 et 2018. Contrairement aux méthodes classiques, YOLO divise l'image en une grille $S \times S$ et prédit simultanément des boîtes englobantes et leurs probabilités de classe pour chaque cellule. Les cadres dont la probabilité dépasse un seuil sont retenus pour localiser les objets.

YOLO se distingue par sa vitesse élevée (jusqu'à 45 FPS sur une Titan X) grâce à une seule passe de l'image dans le réseau CNN. Toutefois, il présente des limitations sur les petits objets ou ceux proches les uns des autres, en raison de sa grille fixe. Son principal avantage reste sa rapidité, ce qui le rend très adapté aux applications temps réel, souvent plus rapide que des méthodes comme Faster R-CNN. [68]

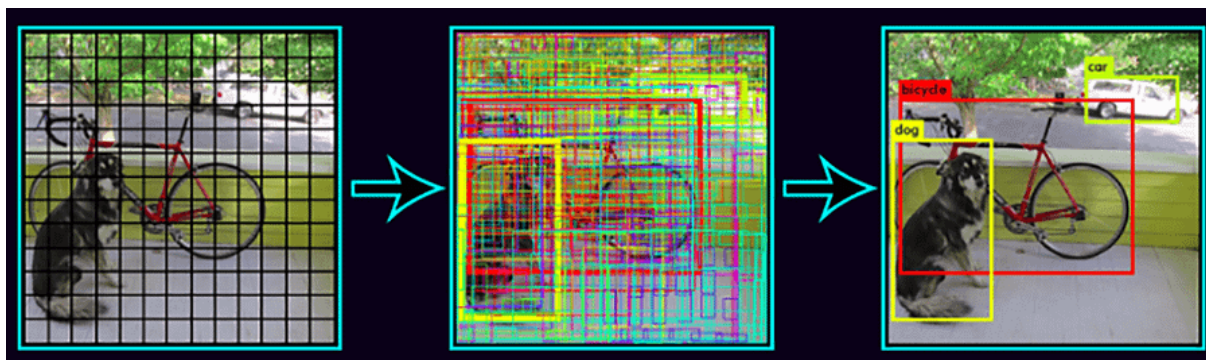


Figure III.13 : Principe de la détection d'objets avec YOLO.

Avantages de YOLO

1. **Détection en temps réel : YOLO** fonctionne en une seule passe (one-shot), ce qui lui permet de détecter des objets très rapidement, adapté aux applications en temps réel comme la surveillance ou les véhicules autonomes.
2. **Pipeline simplifié (end-to-end) :** Le modèle prédit directement les boîtes englobantes et les probabilités de classe dans une seule étape, sans besoin d'un réseau de propositions régionales.
3. **Grande efficacité computationnelle :** En analysant l'image entière d'un coup, YOLO réduit les coûts de calcul et la mémoire nécessaire.
4. **Bonne capacité de généralisation :** YOLO fonctionne efficacement sur différentes classes d'objets et types d'images, même sur des scènes non vues pendant l'entraînement. [8]

Inconvénients de YOLO

1. **Peu précis pour les petits objets :** La division de l'image en grille limite la résolution spatiale pour les petits objets, ce qui peut nuire à leur détection.
2. **Précision de localisation moyenne :** Les objets proches ou superposés peuvent être mal délimités ; les boîtes prédites peuvent être légèrement décalées ou imprécises.

- 3. Moins performant que certains détecteurs deux étapes :** Bien que rapide, YOLO peut être moins précis que Faster R-CNN sur des datasets complexes ou contenant de nombreux petits objets. [8]

III.8.6.1 Le modèle YOLOv2

Il existe plusieurs versions du modèle jusqu'à présent, chaque nouvelle version améliorant la précédente en termes de vitesse et de précision.

Dans le but de développer un modèle plus performant, rapide et robuste, Joseph Redmon et Ali Farhadi ont apporté des améliorations significatives à l'architecture originale de YOLO. Leur nouvelle version intègre Darknet-19 comme réseau de base (backbone) et présente une structure approfondie de 30 couches, comparativement aux 26 couches de YOLO v1.

YOLOv2, est un détecteur d'objets en une seule passe (one-stage) introduit en 2016. Il améliore YOLOv1 grâce à plusieurs innovations :

- L'intégration de batch normalization sur toutes ses couches convolutives pour stabiliser et accélérer l'apprentissage,
- L'emploi du backbone Darknet-19, plus performant,
- L'utilisation de boîtes d'ancrage (anchor boxes) pour prédire différents ratio et tailles d'objets,
- Une stratégie d'entraînement multi-échelle (multi-scale training) permettant de mieux détecter les petits objets. [63]

Grâce à ces améliorations, YOLOv2 atteint une précision élevée (mAP $\sim$ 76–78 % sur les ensembles VOC / COCO) tout en conservant un temps de traitement rapide ($\sim$ 45–67 FPS), surpassant généralement Faster R-CNN et même SSD dans certains cas.

III.8.6.2 Le modèle YOLOv3

YOLOv3 (You Only Look Once version 3), proposé en 2018, est une méthode de détection d'objets rapide et précise, conçue pour fonctionner en temps réel. Elle repose sur l'architecture Darknet-53, composée de 53 couches convolutionnelles, permettant une bonne balance entre vitesse et précision. [5]

YOLOv3 est un détecteur d'objets unifié basé sur CNN qui divise l'image en une grille et prédit plusieurs boîtes englobantes et probabilités par cellule. Chaque prédiction inclut une estimation de présence d'objet et des ajustements de position/taille, fondés sur des anchor boxes optimisées via clustering. Le modèle opère sur trois échelles pour mieux détecter des objets petits et grands, et utilise la suppression non maximale (NMS) pour filtrer les

doublons. Cette organisation permet à YOLOv3 de combiner vitesse et précision dans des applications réelles. YOLOv3 reste très utilisé dans les domaines comme la surveillance ou la conduite autonome, et a inspiré plusieurs évolutions, dont YOLOv4, YOLOv5, et YOLACT, qui visent à optimiser encore plus la précision et la rapidité. [69]

III.8.6.3 Le modèle YOLOv4

YOLOv4, publié en 2020, est une version optimisée du modèle de détection d'objets en temps réel YOLO. Il allie rapidité et précision tout en restant accessible sur un seul GPU classique. Son architecture combine un backbone CSPDarknet53, un neck SPP-PANet pour la fusion multi-échelle, et un head similaire à YOLOv3. Des techniques d'entraînement innovantes comme la Self-Adversarial Training et des améliorations (« Bag of Freebies » et « Bag of Specials ») renforcent sa robustesse. YOLOv4 améliore la précision et la vitesse d'environ 10% et 12% par rapport à YOLOv3. Ce modèle est adapté aux applications exigeant détection rapide et fiable. [15]

III.9 Conclusion

Ce chapitre a présenté un panorama complet des méthodes de segmentation, classification et reconnaissance d'objets dans les images. La segmentation, en tant qu'étape initiale, joue un rôle fondamental en délimitant les régions pertinentes à analyser. Elle facilite les phases ultérieures de détection et de reconnaissance en isolant les objets d'intérêt du fond.

La classification, qu'elle soit supervisée ou non supervisée, permet ensuite d'identifier les caractéristiques propres à chaque classe d'objet. Les méthodes supervisées offrent une grande précision, tandis que les non supervisés sont adaptés aux cas sans annotations.

La reconnaissance d'objet exploite ces caractéristiques pour attribuer des étiquettes aux objets détectés. Les algorithmes modernes, tels que R-CNN, YOLO et SSD, ont permis d'améliorer la rapidité et la précision de la détection. Certains, comme Mask R-CNN, intègrent directement la segmentation dans leur processus. Le choix d'une méthode dépend des contraintes de l'application (temps réel, précision, nature des objets).

Chapitre IV :

Implémentation et

Discussion des résultats

IV.1 Introduction

Dans ce chapitre, nous présentons la phase d'implémentation pratique de notre étude, mettant en œuvre les différentes techniques de traitement d'image et de détection d'objets étudiées précédemment. À travers une série d'expérimentations réalisées sous l'environnement MATLAB. Pour mieux comprendre leur impact, nous avons simulé différents bruits (gaussien, sel et poivre, speckle, poisson, etc.) sur des images, puis appliqué plusieurs filtres de débruitage (moyenneur, gaussien, médian, bilatéral, etc.). Ensuite, nous explorons des algorithmes de détection d'objets basés sur l'apprentissage profond, notamment les modèles YOLOv3 et YOLOv4, appliqués à des images statiques et en temps réel via une webcam. Cette approche intégrée permet d'étudier la performance conjointe du débruitage et de la reconnaissance d'objets dans des conditions variées, simulant des environnements réels.

IV.2 Les images utilisées

Dans le cadre de ce travail, plusieurs images ont été exploitées à des fins de traitement et d'analyse. Certaines de ces images sont intégrées par défaut dans l'environnement MATLAB, tandis que d'autres ont été importées manuellement pour élargir le champ d'expérimentation. Voici une présentation détaillée des images utilisées :

a) **pout.tif (intégrée à MATLAB)**

L'image pout.tif est une image en niveaux de gris de dimensions 291×240 pixels. Cette image est fournie par défaut dans MATLAB et est couramment utilisée dans les démonstrations d'algorithmes de traitement d'image. Elle offre un bon équilibre entre zones uniformes, textures et bords, ce qui en fait un excellent support pour des opérations telles que le filtrage spatial, l'égalisation d'histogramme. Elle a été exploitée notamment pour évaluer l'efficacité de certains filtres d'amélioration de contraste.

b) **cameraman.tif (intégrée à MATLAB)**

L'image cameraman.tif, également intégrée à MATLAB, est une image en niveaux de gris de taille 256×256 pixels. Elle représente un homme manipulant une caméra sur trépied dans un espace extérieur. Cette image est emblématique dans le domaine du traitement d'images et est fréquemment utilisée pour valider visuellement les performances d'algorithmes de filtrage. Sa richesse en détails (structures du manteau, trépied, arrière-plan urbain) en fait une référence pédagogique.

c) perroquet.jpg

L'image perroquet.jpg est une photographie couleur d'un perroquet aux plumes vivement colorées (rouge, bleu, vert). De dimensions 740×1109 pixels, elle a été choisie pour illustrer les traitements spécifiques aux images en couleur, tels que les transformations d'espaces colorimétriques (RGB vers HSV, YCbCr, etc.), la segmentation par couleur, et l'analyse chromatique. La richesse spectrale de cette image en fait un cas d'étude pertinent pour les applications liées à la vision par ordinateur.

d) la_rose.jpg

L'image la_rose.jpg est une photographie artistique en niveaux de gris d'une rose, de dimensions 1034×1550 pixels. Son rendu monochrome met en évidence la finesse des contours et des textures florales, ce qui la rend particulièrement adaptée à des traitements tels que la détection de bords, la mise en relief.



(a)



(b)



(c)



(d)

Figure IV.1: (a) pout de taille 291×240. (b) cameraman de taille 256×256. (c) perroquet de taille 740×1109. (d) la_rose de taille 1034x1550

IV.3 RGB to GRAY

Dans notre projet on va travailler uniquement sur les images en niveau de gris donc nous avons été nécessaire de convertir l'image en niveau de gris.

IV.3.1 Application MATLAB (exemple perroquet)

La commande utilisée c'est = **rgb2gray**

Par exemple voilà le script sur Matlab qui permet d'utiliser :

```
perroquet = imread("perroquet.jpg"); % Charger l'image (remplacez par votre chemin)
figure ;
imshow(perroquet)
perroquet_gray = rgb2gray(perroquet);
figure,
imshow(perroquet_gray)
% Pour vérifier le résultat :
size(perroquet) % Par exemple [1109 740 3] → image RGB ; [1109 740] → image en gris
```

IV.3.2 Résultat obtenu

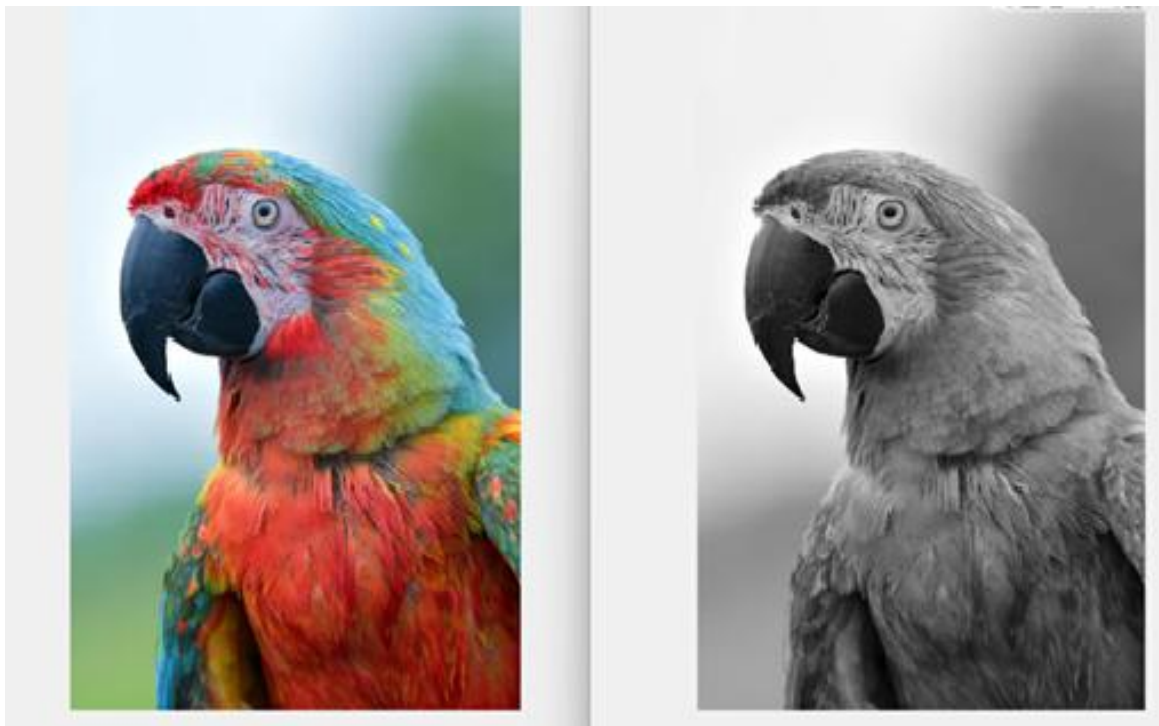


Figure IV.2 : Image de perroquet en RGB transférer en gris

IV.4 Histogramme d'image

En utilisant la fonction **imhist** : (exemple pout.tif de **Figure IV.1 : (a)**)

```
im_in = imread("pout.tif");  
a = min(im_in(:))  
b = max(im_in(:))  
im_out = histeq(im_in);  
figure,  
subplot(2,2,3), imshow(im_in);  
subplot(2,2,4), imshow(im_out);  
subplot(2,2,1), imhist(im_in);  
subplot(2,2,2), imhist(im_out);
```

IV.4.1 Résultat obtenu

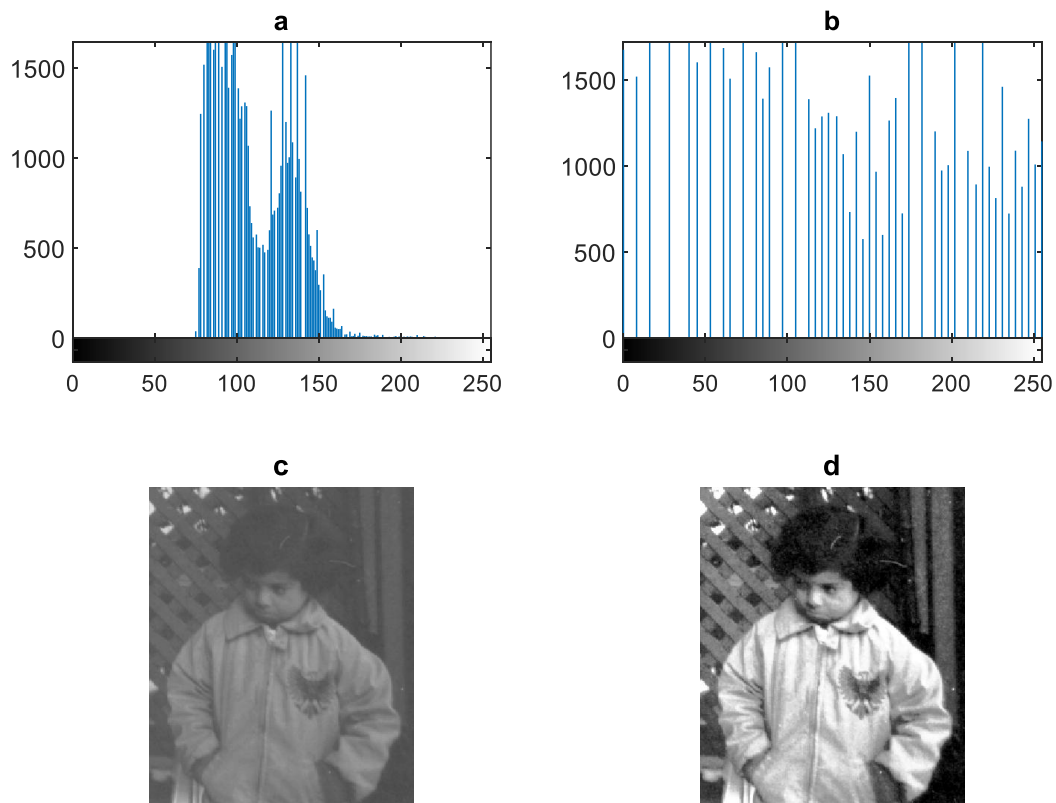


Figure IV.3 : (a) histogramme d'image. (b) histogramme d'image après l'augmentation de contraste. (c) image pout original. (d) image après l'augmentation de contraste.

IV.4.2 Description de la fonction ‘imhist’

La fonction ‘**imhist**’ de MATLAB constitue un outil fondamental pour l'analyse statistique des images numériques. Cette fonction calcule et affiche l'histogramme d'une image en niveaux de gris, révélant la distribution des intensités lumineuses sur l'ensemble des pixels. L'histogramme représente graphiquement le nombre d'occurrences de chaque niveau de gris, de 0 (noir) à 255 (blanc) pour les images 8 bits standard.

IV.4.3 Analyse des résultats

IV.4.3.1 Implémentation et paramètres techniques

Le code présenté une implémentation basique mais efficace de l'analyse histogrammique. L'instruction **im_in = imread("pout.tif")** charge l'image d'entrée, tandis que **im_out = imhist(im_in)** génère les données de l'histogramme. Les variables **a = min(im_in(:))** et **b = max(im_in(:))** permettent d'identifier respectivement les valeurs minimale et maximale de luminance présentes dans l'image ce qui aide à comprendre si l'image utilise bien toute l'échelle des niveaux de gris (du noir au blanc).

IV.4.3.2 Interprétation des résultats obtenus

L'analyse comparative des deux histogrammes révèle des caractéristiques distinctes de distribution tonale. L'histogramme de gauche présente une concentration marquée des pixels dans les basses fréquences (zones sombres), avec un pic significatif autour des valeurs 50-100, indiquant une image globalement sous-exposée ou à dominante sombre. Dans le second histogramme (b), les pixels sont répartis plus largement sur toute l'échelle de gris, suggérant une meilleure utilisation de la dynamique tonel disponible et un contraste plus équilibré.

IV.5 Les méthodes de bruitage

IV.5.1 Bruit gaussien

La commande utilisée ç'est : **imnoise(x,'gaussian',m, σ);**

Ajoute du bruit blanc gaussien avec une moyenne **m** et une variance **σ** .

On peut utiliser : **Image + sigma_noise * randn(size(Image));**

IV.5.1.1 Script utilisé

```
clc
i = imread("la_rose.jpg");
im1 = rgb2gray(i);
x=im1;
I_gaussian = imnoise(x,'gaussian',0,7/255); %bruit additif
imshow(I_gaussian)
```

IV.5.1.2 Résultat obtenu

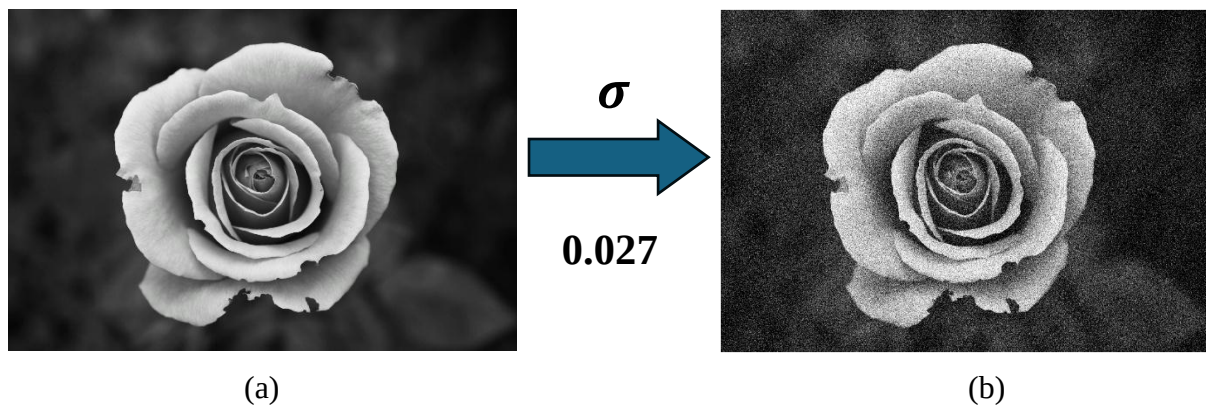


Figure IV.4 : (a) image originale. (b) représente la dégradation d'image avec bruit gaussien.

IV.5.1.3 Discussion de résultat

L'image (a) est l'image originale en niveaux de gris. L'image (b) montre l'ajout d'un bruit gaussien avec une moyenne de 0 et une variance de 0.027, ce qui rend l'image plus floue et irrégulière.

Note :

Malgré qu'on ait remarqué que la rose est une image en niveau de gris **Figure IV.4** mais elle est téléchargée sur le web automatiquement à la forme de 3 couleurs (RGB) c'est pour ça on peut voir le bruit colorer si on n'a pas transformé à la forme de noir et blanc.

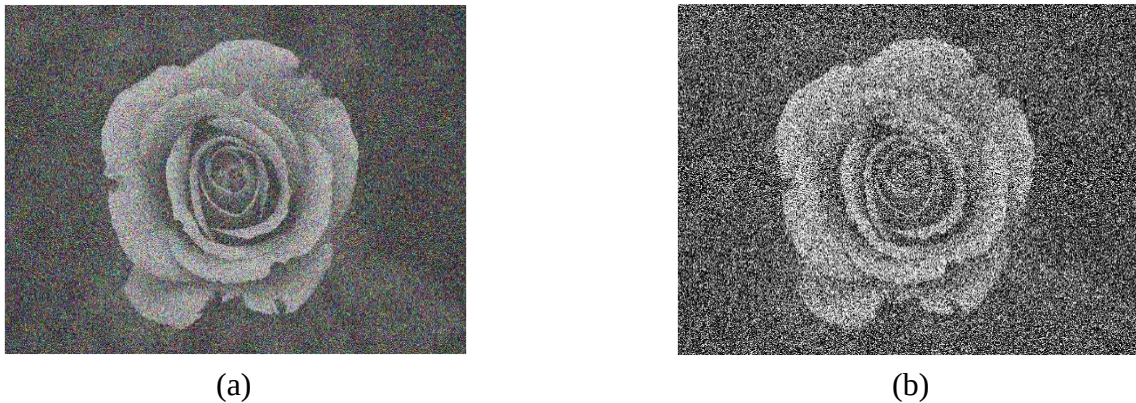
Exemple :

Figure IV.5 : (a) image en RGB avec bruit gaussien. (b) image ou niveau de gris avec bruit gaussien.

IV.5.2 Bruit Spackle

La commande utilisée ç'est : **imnoise(l,'speckle',var_speckle)**

Ajoute du bruit multiplicatif avec une variance speckle (σ).

IV.5.2.1 Script utilisé

```
clc
clear all
im1 = imread(" la_rose.png");
i = rgb2gray(im1);
figure, imshow(i)
x=i;
I_speckle = imnoise(x, 'speckle',0.09);    % Bruit multiplicative
figure, imshow(I_speckle)
```

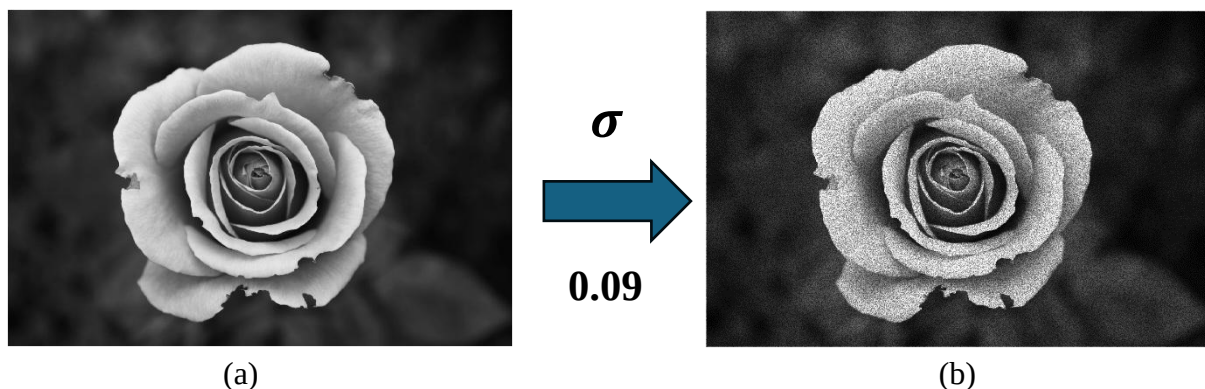
IV.5.2.2 Résultat obtenu

Figure IV.6 : (a) image ou niveau de gris. (b) représente la dégradation d'image avec bruit speckle de variance 0.09.

IV.5.2.3 Discussion de résultat

L'application du bruit speckle avec une variance $\sigma = 0.09$ sur l'image de la rose démontre clairement les caractéristiques multiplicatives de ce type de bruit. Les résultats montrent que les zones sombres sont moins affectées que les zones claires, permettant une meilleure préservation des contours et structures importantes comparativement aux autres types de bruit. Cette particularité confirme l'importance d'utiliser des techniques de débruitage adaptatives spécifiquement conçues pour traiter la nature multiplicative du bruit speckle.

IV.5.3 Bruit de Sel et Poivre (Salt and Pepper)

La commande utilisée ç'est : `imnoise(x, 'salt & pepper', d);`

Où d correspond à la densité du bruit. Cela affecte environ $d \cdot \text{numel}(x)$ pixels.

IV.5.3.1 Script utilisé

```
i = imread('cameraman.tif');  
figure, imshow(i);  
x=i;  
I_sel_poivre = imnoise(x, 'salt & pepper', 0.03); % Bruit impulsionnel (multiplicatif)  
figure  
imshow(I_sel_poivre)
```

IV.5.3.2 Résultat obtenu

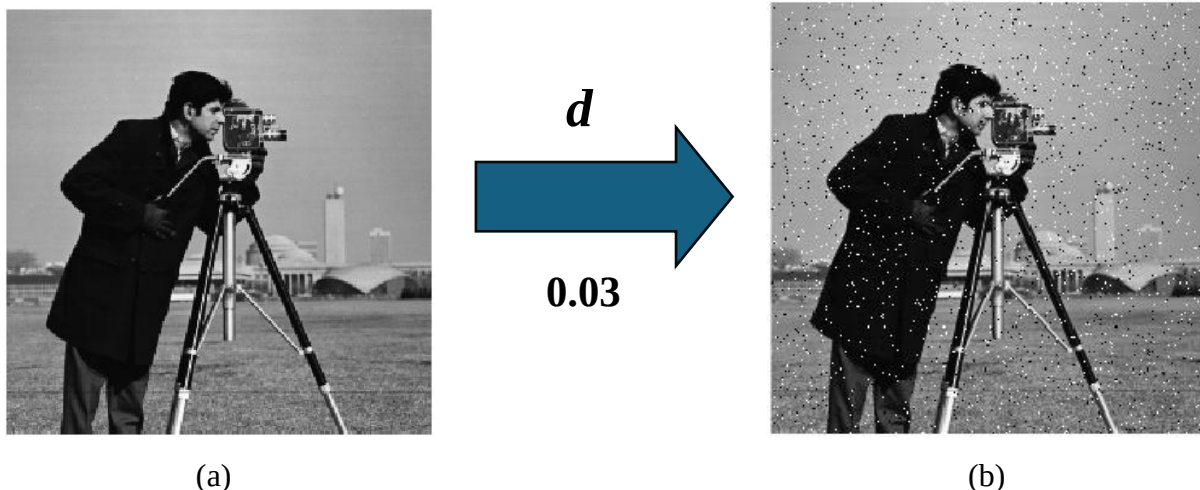


Figure IV.7 : (a) image originale. (b) image avec bruit de Sel et Poivre.

IV.5.3.3 Discussion de résultat

D'après la **Figure IV.7** l'image (a) est l'image d'origine sans bruit. L'image (b) montre l'ajout d'un bruit de type Sel et Poivre avec une densité de 0.03, ce qui fait apparaître des points noirs et blancs aléatoires.

IV.5.4 Flou de mouvement (Motion Blur)

La commande utilisée ç'est : **fspecial('motion',lon ,theta)**

Ou **lon** spécifie la longueur du mouvement et **theta** spécifie l'angle de mouvement en degrés dans le sens inverse des aiguilles d'une montre.

IV.5.4.1 Script utilisé

```
I_original = imread("perroquet_gray.png");  
PSF = fspecial('motion', 20, 30); % Créer un flou de mouvement (20 pixels, 30 degrés)  
blurred = imfilter(I_original, PSF, 'conv', 'circular');  
figure, imshow(blurred)
```

IV.5.4.2 Résultat obtenu

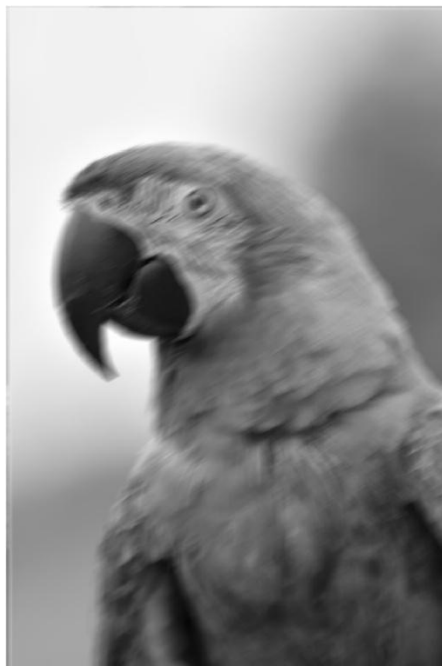


Figure IV.08 : Perroquet traité avec flou de mouvement.

Remarque :

Une fois convolué avec une image, le mouvement linéaire d'une caméra Le filtre devient un vecteur pour les mouvements horizontaux et verticaux.

IV.5.5 Bruit de poisson (bruit photonique)

La commande utilisée ç'est : **I = imnoise(i,'poisson');**

IV.5.5.1 Scripte utilisé

```
I = imread('cameraman.tif');  
p = imnoise(I,'poisson');  
figure, imshow(p)  
I2 = im2double(imread('cameraman.tif')); %Image en fonction double;  
k = imnoise(I2,'poisson');  
figure, imshow(k)
```

IV.5.5.2 Affichage de résultat

Figure IV.9 : image en classe uint8 avec bruit de poisson.

Remarque :

Le bruit de poisson ne fonction pas dans les images en class double car il suivie l'intensité d'image pour la modification de bruit.

IV.6 Différence entre bruit gaussien et bruit speckle

IV.6.1 Distribution spatiale

Le bruit gaussien affecte tous les pixels de manière uniforme et indépendante, ce qui donne un effet de grain régulier sur toute la surface de l'image. La variance du bruit reste constante quelle que soit la zone de l'image considérée.

Le bruit speckle, quant à lui, présente une intensité variable selon les régions : les zones claires sont plus fortement dégradées que les zones sombres, Cela crée une texture irrégulière qui suit un peu mieux les formes de l'image.

IV.6.2 Effet visuel

Visuellement, le bruit gaussien réduit globalement le contraste et la netteté de l'image, créant un effet de flou généralisé qui masque uniformément les détails fins.

Le bruit speckle donne un effet de granulation plus marqué, mais il garde mieux les contours et les formes importantes, surtout dans les zones sombres, car son intensité diminue naturellement dans les zones sombres où se trouvent souvent les contours significatifs.

IV.6.3 Méthodes de traitement

Comme les deux bruits sont différents, on ne les traite pas de la même manière.

Le bruit gaussien répond efficacement aux filtres linéaires classiques comme les filtres moyenneurs, gaussiens.

Le traitement du bruit speckle nécessite des filtres adaptatifs plus sophistiqués comme les filtres de Lee ou Frost, qui tiennent compte de sa nature multiplicative et de sa dépendance aux statistiques locales de l'image.

IV.7 Les filtre adaptatif

IV.7.1 Filtre de Wiener

Le filtre de Wiener particulièrement utile dans le débruitage d'images affectées par un bruit gaussien blanc ou par des effets de flou modérés, tout en maintenant un bon compromis entre lissage et préservation des détails.

IV.7.1.1 Scripte utilisé

```
I_original = im2double(imread('cameraman.tif'));
PSF = fspecial('motion', 21, 11);
blurred = imfilter(I_original, PSF, 'conv', 'circular'); % Ajouté le flou sur I_original
% Créer de bruit gaussien :
noise_var = 0.001;
blurred_noisy = imnoise(blurred, 'gaussian', 0, noise_var);
% Estimation du rapport signal/bruit
estimated_nsr = noise_var / var(I_original(:));
wnr2 = deconvwnr(blurred_noisy, PSF, estimated_nsr);
psnr_blurred = psnr(blurred, I_original);
psnr_restored = psnr(wnr2, I_original);
% Affichage des résultats :
fprintf('Blurred PSNR: %.2f dB\n', psnr_blurred);
fprintf('restored PSNR: %.2f dB\n', psnr_restored);
fprintf('Amélioration PSNR: %.2f dB\n', psnr_restored - psnr_blurred);
subplot(1,3,1);
imshow(I_original);
title('Image original');
subplot(1,3,2);
imshow(blurred_noisy);
title('Image avant filtrage');
subplot(1,3,3);
imshow(wnr2);
title('Image après filtrage Wiener');
```

Bute de ce programme :

Ce code simule un flou de mouvement et un bruit gaussien sur une image, puis applique un filtre Wiener pour restaurer l'image. Il affiche les résultats visuellement et mesure quantitativement l'amélioration via le PSNR. C'est un exemple est efficace pour comprendre le traitement d'image par déconvolution Wiener dans MATLAB.

IV.7.1.2 Affichage de figure



Figure IV.10 : image de cameraman restauré par le filtre de Wiener.

IV.7.1.3 Analyse des résultats

```
Blurred PSNR: 19.29 dB
restored PSNR: 19.83 dB
Amélioration PSNR: 0.53 dB
fx >> |
```

Le NSR (Noise-to-Signal Ratio) est estimé en divisant la variance du bruit par la variance du signal original.

Cette estimation est importante pour que le filtre Wiener puisse équilibrer la suppression du bruit et la restauration du flou.

Aspects positifs :

Récupération des détails principaux (silhouette, équipement).

Réduction notable du flou de mouvement.

Préservation du contraste général.

Limitations visibles :

Présence de bruit résiduel (visible dans l'image restaurée).

Certains détails fins ne sont pas complètement récupérés.

IV.7.2 Filtre Médian

La commande utilisée ç'est : **medfilt2(I_bruitee, [n n]);**

n = nombre binaire.

IV.7.2.1 Script utilisé

```
clc
clear all
I_original = imread("perroquet.jpg");
I = rgb2gray(I_original);
I_bruitee = imnoise(I, 'salt & pepper', 0.05); % Image bruitée impulsionnellement
I_filtre = medfilt2(I_bruitee, [3 3]); % Filtre médian avec fenêtre 3x3
imshowpair(I_bruitee, I_filtre, 'montage');
title('Image bruitée (gauche) et image filtrée par filtre médian (droite)');
```

IV.7.2.2 Résultat obtenu



Figure IV.11 : Image de perroquet avec bruit Sel et Poivre traité par le filtre de médian.

IV.7.2.3 Analyse des résultats

D'après notre observation de la **figure IV.11** on a remarqué que :

L'expérimentation utilise un bruit "salt & pepper" de densité 5% sur une image de perroquet, traité par un filtre médian 3×3 . Ce choix est presque optimal car il associe le type de bruit impulsionnel avec le filtre le plus adapté à sa correction.

La comparaison visuelle révèle une élimination presque complète du bruit impulsionnel. L'image filtrée présente une surface parfaitement lisse sans pixels parasites, démontrant l'efficacité remarquable du filtre médian contre ce type de dégradation spécifique.

Le filtre médian maintient intégralement la netteté des contours et textures du perroquet. Les détails fins comme les plumes, l'œil et le bec conservent leur définition originale, contrairement aux filtres linéaires qui introduiraient un flou généralisé.

IV.7.2.4 Implications pratiques et recommandations

Ces résultats établissent le filtre médian comme référence pour le traitement du bruit impulsionnel. La taille de fenêtre 3×3 s'avère optimale pour une densité de bruit de 5%, fournissant un guide pratique pour des applications similaires.

IV.7.3 Filtre bilatéral

La commande utilisée c'est : **imbilatfilt(I_noisy, sigma_intensity, sigma_spatial)**

IV.7.3.1 Scripte utilisé

```
I = imread("perroquet_gray.png");
I = rgb2gray(I); % Conversion en niveaux de gris
I = double(I) / 255; % Normalisation entre 0 et 1
%% 2. Ajout du bruit gaussien
sigma_noise = 0.05; % Variance du bruit
I_noisy = I + sigma_noise * randn(size(I));
I_noisy = max(0, min(1, I_noisy)); % Limitation entre 0 et 1
%% 3. Application du filtre bilatéral
sigma_spatial = 2; % Paramètre spatial (ps)
sigma_intensity = 0.1; % Paramètre d'intensité (pi)
I_bilateral = imbilatfilt(I_noisy, sigma_intensity, sigma_spatial);
subplot(1, 3, 1);
imshow(I);
% Image bruitée
subplot(1, 3, 2); imshow(I_noisy);
% Filtre bilatéral
subplot(1, 3, 3);
imshow(I_bilateral);
```

IV.7.3.2 Résultat obtenu

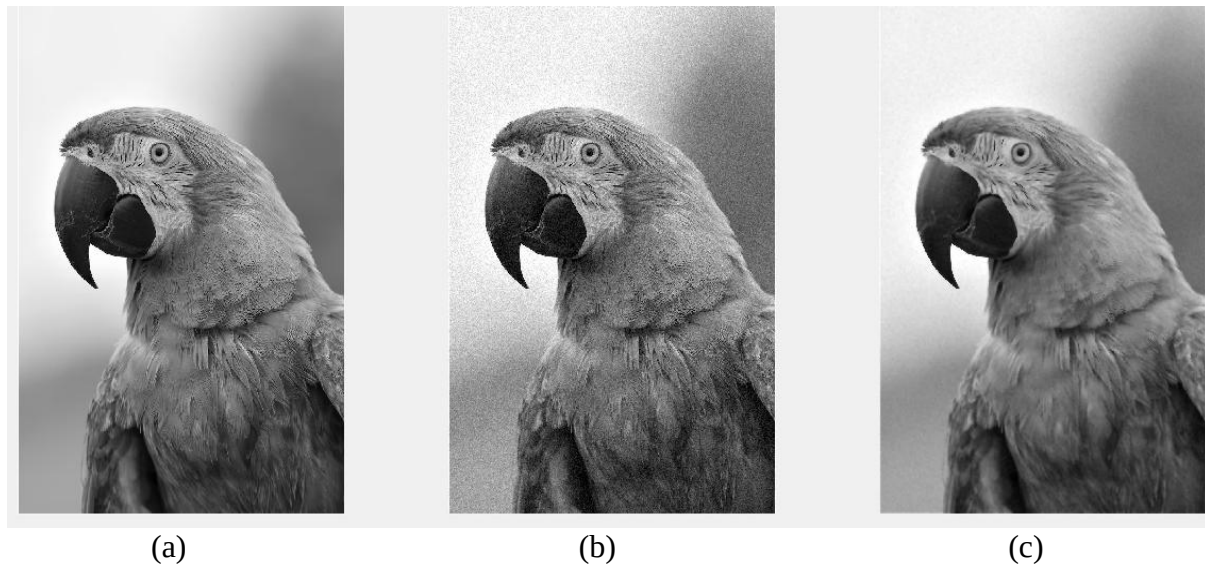


Figure IV.12 : (a) image originale. (b) image avec bruit gaussien. (c) image traitée par le filtre bilatéral.

IV.7.3.3 Analyse des résultats

D'après la **Figure VI.12** dans l'image du milieu, on voit que le bruit gaussien rend l'image floue avec une perte de détails, surtout sur les plumes du perroquet. Après application du filtre bilatéral (image de droite), on voit clairement que le bruit a diminué, surtout dans les zones uniformes comme le fond. Contrairement à d'autres filtres plus simples comme le filtre moyennneur ou gaussien, le filtre bilatéral ne floute pas les contours : les détails du bec et des plumes du perroquet restent visibles et nets. Ce filtre est donc efficace car il enlève le bruit sans trop toucher aux détails importants de l'image.

IV.8 Filtre non adaptatif

IV.8.1 Filtre moyennneur

La commande utilisée ç'est : **fspecial('average', [n n]);**

IV.8.1.1 Scripte utilisé

```
I_original = imread("la_rose.jpg");
I_gray = im2gray(I_original);
I_bruitee = imnoise(I_gray, 'gaussian', 0, 0.02);
M = fspecial('average', [3 3]); % filtre moyennneur avec fenêtre de 3x3
I_filtered = imfilter(I_bruitee, M);
imshowpair(I_bruitee, I_filtered, 'montage');
title('Image bruitée (gauche) et image filtrée par filtre moyennneur (droite)');
```

IV.8.1.2 Résultat obtenu

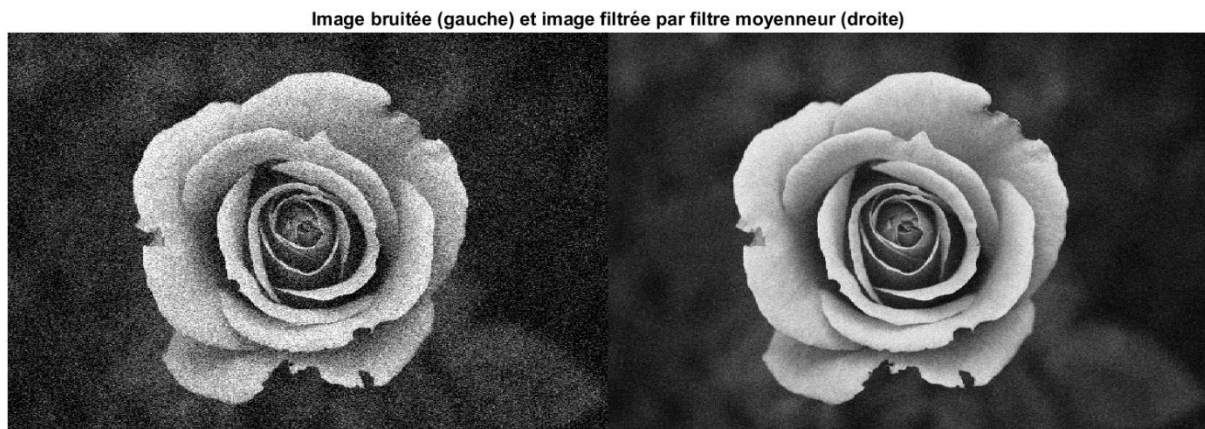


Figure IV.13 : Image de la rose bruitée et traitée par le filtre de moyenneur.

IV.8.1.3 Analyse des résultats

L'application du filtre moyenneur de taille 3×3 sur l'image bruitée par du bruit gaussien (variance $\sigma = 0.02$) démontre l'efficacité de cette approche de débruitage linéaire. L'image de gauche présente le bruit gaussien caractéristique sous forme de grain uniforme réparti sur toute la surface, dégradant significativement la qualité visuelle et masquant les détails fins de la rose. L'image de droite, après filtrage, montre une amélioration notable de la qualité avec une réduction considérable du bruit et une restauration partielle de la netteté.

Le filtre moyenneur fonctionne selon le principe de convolution où chaque pixel est remplacé par la moyenne de ses voisins dans une fenêtre 3×3 . Cette opération lisse efficacement les variations aléatoires du bruit gaussien tout en préservant les structures principales de l'image. Cependant, on observe également un léger effet de flou sur les contours.

IV.8.2 Filtre de gaussien

La commande utilisée c'est : **imgaussfilt(I_bruitee,sigma);**

IV.8.2.1 Scripte utilisé (exemple de la rose appliqu'ont en bruit gaussien)

```
I = rgb2gray(imread("la_rose.jpg"));
I_bruitee = imnoise(I,"gaussian",0,0.07);
I_gauss = imgaussfilt(I_bruitee,2);      % sigma = 2
imshowpair(I_bruitee, I_gauss, 'montage');
title('Image originale (gauche) et image filtrée par filtre gaussien (droite)');
```

IV.8.2.2 Résultat obtenu

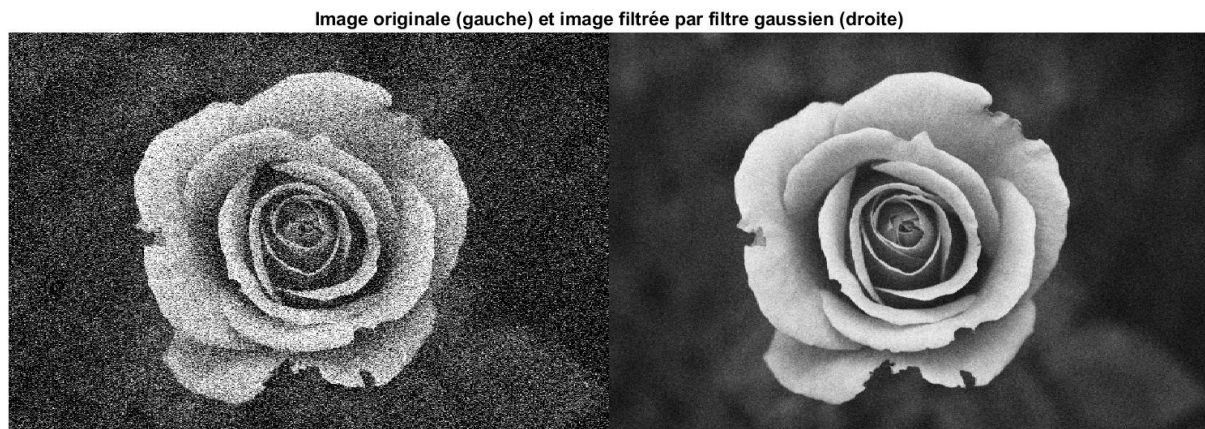


Figure IV.14 : Image de la rose avec bruit de gaussien traitée par le filtre de gaussien.

IV.8.2.3 Analyse des résultats

L'application du filtre gaussien sur l'image de gauche avec $\sigma = 2$ affectées par un bruit gaussien de variance 0.07 démontre une efficacité remarquable dans la réduction du bruit, tandis que l'image filtrée (droite) révèle une amélioration spectaculaire de la qualité visuelle. Le filtre gaussien réussit à éliminer la majorité du bruit tout en restaurant la lisibilité générale de l'image.

IV.8.2.3.1 Caractéristiques du filtrage

Le filtre gaussien, contrairement au filtre moyenneur, utilise une pondération basée sur la fonction gaussienne qui privilégie le pixel central et diminue progressivement l'influence des pixels périphériques. Cette approche produit un lissage plus naturel et moins brutal que le filtre moyenneur, expliquant la qualité supérieure des résultats obtenus. Le paramètre $\sigma = 2$ offre un bon compromis entre réduction du bruit et préservation des structures principales de l'image.

IV.8.2.3.2 Compromis netteté-débruitage

Même si le débruitage avec le filtre est efficace, on remarque que certains contours deviennent plus flous et que des détails fins disparaissent, surtout dans les zones avec des textures délicates comme les pétales. Cela s'explique par le fonctionnement du filtre gaussien, qui lisse l'image.

En réduisant les changements rapides de luminosité. Du coup, il enlève le bruit mais aussi certains petits détails importants.

IV.9 Détection et Reconnaissance d'objet

IV.9.1 YOLOv3

Scripte utilisé :

```
% Charger une image
img = imread("image.jpg");
% Charger le modèle YOLOv3 (après avoir installé le support)
model = yolov3ObjectDetector('darknet53-coco');
% Faire la détection :
[bboxes, scores, labels] = detect(model, img);
% Afficher les résultats :
result = insertObjectAnnotation(img, 'rectangle', bboxes, cellstr(labels));
imshow(result);
```

Résultat obtenu :



Figures IV.15 : Détection d'objets à l'aide de YOLOv3. (a) Image originale, (b) Résultat de la détection avec annotations des objets détectés.

IV.9.1.1 Analyse des résultats

La figure montre que le modèle YOLOv3 a bien détecté plusieurs objets dans une rue : des personnes, des sacs, des feux de signalisation, etc. Chaque objet est entouré d'un rectangle avec son nom (ou étiquette).

Même si plusieurs personnes sont proches ou se croisent, le modèle arrive à les détecter correctement. Cela montre que YOLOv3 est efficace même quand l'image est complexe.

IV.9.2 YOLOv4

Script utilisé :

```
detector = yolov4ObjectDetector('csp-darknet53-coco');  
% Lire une image  
img = imread("image.jpg");  
% Faire la détection  
[bboxes, scores, labels] = detect(detector, img);  
% Afficher les résultats  
detectedImg = insertObjectAnnotation(img, 'rectangle', bboxes, cellstr(labels));  
imshow(detectedImg);
```

Résultat obtenu :



Figures IV.16 : Résultat de YOLOv4 avec annotations des objets détectés

IV.9.2.1 Analyse des résultats

Dans notre expérience, Le modèle YOLOv4 n'a pas donné les performances escomptées. Bien que le modèle ait pu détecter un certain nombre d'objets dans l'image (comme des personnes et quelques sacs), plusieurs objets clairement visibles ont été ignorés ou mal localisés.

IV.9.3 Comparaison entre les deux model

Dans notre scénario d'application, YOLOv3 offre de meilleures performances que YOLOv4, malgré une architecture plus ancienne. Il se montre plus précis, cohérent et fiable, notamment dans des scènes denses. Cette observation montre que le choix d'un modèle ne dépend pas

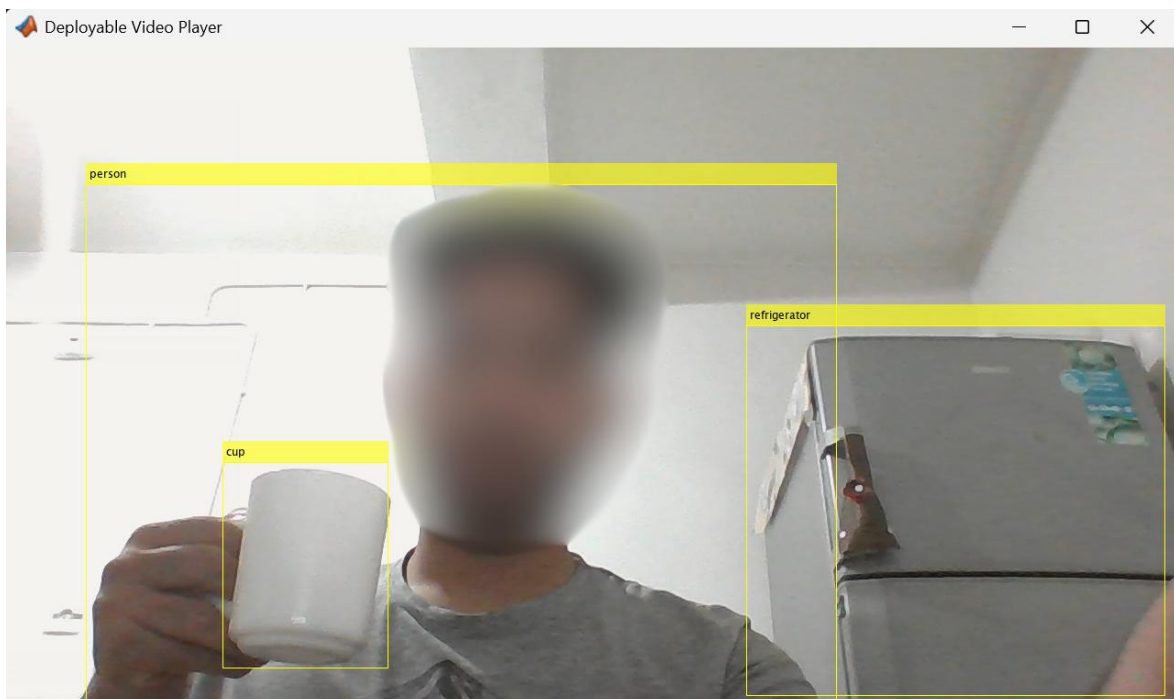
uniquement de sa nouveauté ou de ses promesses théoriques, mais aussi de son adéquation avec les données et les conditions spécifiques de l'application.

IV.9.4 WEBCAM

Code utilise :

```
% Charger le détecteur YOLOv3 pré-entraîné (sur COCO)
detector = yolov3ObjectDetector('tiny-yolov3-coco'); % ou 'darknet53-coco'
% Initialiser la webcam
cam = webcam; % Créer une fenêtre de visualisation
videoPlayer = vision.DeployableVideoPlayer(); % Boucle de détection
while true
    img = snapshot(cam); % Capturer une image de la webcam
    % Appliquer la détection d'objets
    [bboxes, scores, labels] = detect(detector, img);
    % Annoter les objets détectés
    annotatedImg = insertObjectAnnotation(img, 'rectangle', bboxes, cellstr(labels));
    step(videoPlayer, annotatedImg); % Afficher le résultat
end
```

Résultat obtenu :



Figures IV.17 : Résultat de détection d'objets en temps réel avec YOLOv3.

IV.9.5.1 Qualité des detections

- Les boîtes englobantes apparaissent en jaune, et les labels sont affichés en haut à gauche des boîtes, ce qui est standard pour la visualisation d'objets détectés.
- Toutes les trois classes d'objets sont bien différenciées sans chevauchement problématique des boîtes.
- Il n'y a pas d'objets détectés en double ou d'erreurs évidentes visibles sur cette image.

IV.9.5.2 Analyse le résultat

Ce résultat est un bon exemple de détection multi classes efficace réalisée par un modèle de type YOLOv3 appliqué sur la webcam. Les classes d'objets sont bien détectées avec localisation correcte, et l'annotation visuelle est claire et intuitive.

VI.10 Conclusion

L'étude expérimentale présentée dans ce chapitre a permis d'évaluer l'impact des différentes techniques de débruitage sur la qualité des images, et leur influence sur les performances de détection d'objets. Les résultats montrent que les filtres adaptatifs comme le filtre de Wiener ou le filtre bilatéral offrent un bon compromis entre réduction de bruit et préservation des détails, tandis que le filtre médian reste particulièrement efficace contre le bruit impulsionnel. Concernant la détection d'objets, YOLOv3 s'est avéré plus performant que YOLOv4 dans notre contexte d'utilisation, notamment en termes de précision et de robustesse. Enfin, l'implémentation en temps réel avec webcam a démontré la faisabilité d'une application pratique de ces techniques combinées. Ces observations ouvrent la voie à des améliorations futures dans les systèmes de vision artificielle embarqué.

Conclusion générale

Au terme de ce travail, nous avons mis en évidence l'importance du traitement d'images numériques dans l'amélioration de la qualité visuelle et l'extraction d'informations pertinentes. Nous avons d'abord étudié les fondements de l'image numérique, ses caractéristiques essentielles ainsi que les différentes formes de bruit susceptibles de la dégrader. Ensuite, nous avons présenté plusieurs techniques de filtrage, en distinguant les méthodes non adaptatives, simples mais limitées, et les méthodes adaptatives, plus performantes car elles tiennent compte du contexte local de l'image. Enfin, nous avons abordé la détection et la reconnaissance d'objets, qui représentent un champ d'application majeur du traitement d'images, en particulier avec l'intégration des approches modernes basées sur l'apprentissage profond.

Les résultats obtenus soulignent que le choix du filtre ou de la méthode de reconnaissance doit être adapté à la nature du bruit, à la complexité de la scène et aux objectifs visés. Les approches adaptatives et les techniques d'intelligence artificielle constituent aujourd'hui des solutions prometteuses pour améliorer la robustesse et la précision des systèmes de vision par ordinateur. Ce mémoire ouvre ainsi la voie à des perspectives de recherche futures, notamment dans l'optimisation des algorithmes hybrides combinant débruitage et reconnaissance en temps réel, afin de répondre aux besoins croissants des applications industrielles et scientifiques.

ANNEXES

Annexe A

Explication du code MATLAB pour le filtrage Wiener d'une image floutée et bruitée

1. Chargement et conversion de l'image originale

```
I_original = im2double(imread('cameraman.tif'));
```

- `imread` charge l'image « cameraman.tif ».
- `im2double` convertit l'image en format double précision avec des valeurs normalisées entre 0 et 1, ce qui est recommandé pour les opérations de traitement d'image.

2. Création d'un flou de mouvement

```
PSF = fspecial('motion', 21, 11);
```

```
blurred = imfilter(I_original, PSF, 'conv', 'circular');
```

- `fspecial('motion', 21, 11)` crée une fonction de flou (PSF = Point Spread Function) simulant un flou de mouvement de 21 pixels à un angle de 11 degrés.
- `imfilter` applique ce flou à l'image originale, avec une convolution circulaire pour gérer les bords.

3. Ajout de bruit

```
noise_var = 0.0001;
```

- On ajoute un bruit blanc gaussien de variance 0.0001 à l'image floutée pour simuler un bruit réaliste.

```
blurred_noisy = imnoise(blurred, 'bruitee', 0, noise_var);
```

- `imnoise` : ajouté bruit gaussien, speckle, poisson et sel et poivre.

On ajoute un bruit blanc gaussien de variance 0.0001 à l'image floutée pour simuler un bruit réaliste.

4. Estimation du rapport signal sur bruit (NSR)

```
estimated_nsr = noise_var / var(I_original(:));
```

- Le NSR (Noise-to-Signal Ratio) est estimé en divisant la variance du bruit par la variance du signal original.
- Cette estimation est importante pour que le filtre Wiener puisse équilibrer la suppression du bruit et la restauration du flou.

5. Application du filtre de Wiener inverse

`wnr2 = deconvwnr(blurred_noisy, PSF, estimated_nsr);`

- `deconvwnr` applique la déconvolution Wiener sur l'image bruitée et floutée, en utilisant la PSF et le NSR estimé.
- Le résultat est une image restaurée, où le flou et le bruit sont atténués.

7. Affichage des images

`subplot(n,m,x);`

La fonction `subplot` permet de créer plusieurs graphiques dans une seule fenêtre en divisant la zone d'affichage en une grille.

Paramètres :

- **n** : Nombre de lignes de la grille
- **m** : Nombre de colonnes de la grille
- **x** : Position du sous-graphique actuel (numérotation de 1 à $n \times m$)

`imshow('')` : est une fonction utilisée pour afficher des images.

`title('')` : est utilisée pour ajouter un titre à une figure ou une image affichée dans une fenêtre graphique.

Histogramme :

Histeq (I) : transforme l'image en niveaux de gris I de sorte que l'histogramme de l'image en niveaux de gris de sortie J comporte 64 cases et soit approximativement plat.

ANNEXE B

L'algorithme K-moyens (ou K-means) :

```
I = imread('cameraman.tif'); % Charger une image couleur ou niveaux de gris
k = 2;                       % Nombre de clusters
L = imsegkmeans(I, k);       % Segmentation k-means
imshow(labeloverlay(I, L)); % Affiche l'image avec les segments colorés
title('Segmentation par k-means');
```

L'algorithme fuzzy c-moyens (ou Fuzzy C-means) :

```
I = imread('cameraman.tif');
I = double(I);
% Étape 2 : Mise en forme des données
data = I(:); % On transforme l'image en vecteur (1D) pour FCM
% Étape 3 : Appliquer Fuzzy C-Means (FCM)
nombre_de_classes = 3; % Choisissez le nombre de clusters souhaité
[centres, U] = fcm(data, nombre_de_classes); % centres = centres des classes, U = degré
d'appartenance
% Étape 4 : Attribuer chaque pixel à une seule classe
[~, classe_max] = max(U); % Trouver la classe avec le degré d'appartenance maximal
image_segmentee = reshape(classe_max, size(I)); % Reformuler en image
% Étape 5 : Affichage des résultats
figure;
imshow(label2rgb(image_segmentee)); % Affichage en couleurs
title('Image segmentée avec FCM');
```

Seuillage :

```
I = imread('La_rose'); % Charger une image

Igray = rgb2gray(I); % Convertir en niveaux de gris si couleur
IB = imbinarize(Igray, 0.4); % Seuillage avec seuil 0.5
figure, imshow(IB);
figure, imshow(I)
```

Seuillage avec Otsu :

```
level = graythresh(Igray);      % Calcul automatique du seuil avec Otsu (niveau entre 0 et 1)
IB = imbinarize(Igray, level);  % Binarisation avec ce seuil
```

Dilatation :

```
I = imread('cameraman.tif');
% 2. Conversion en binaire avec un seuil (ici 0.5)
IB = imbinarize(I, 0.5);
% 3. Créer un élément structurant (ex : disque de rayon 3)
SE = strel('disk', 3);
% 4. Appliquer la dilatation
IB_dilated = imdilate(IB, SE);
figure;
subplot(1,2,1); imshow(IB); title('Image binaire originale');
subplot(1,2,2); imshow(IB_dilated); title('Après dilatation');
```

L'érosion :

```
SE = strel('disk', 3);
IB_eroded = imerode(IB, SE);
```

Ouverture & Fermeture :

```
% Charger ou créer une image binaire exemple
img = imbinarize(rgb2gray(imread('C:\Users\akram\OneDrive\Pièces
jointes\Imágenes\images mimore\perroquet_gray.png'))); % Autre exemple si l'image est en
niveaux de gris
% Créer un élément structurant (ici un disque de rayon 5 pixels)
se = strel('disk',5);
% Ouverture : érosion suivie de dilatation
img_ouverture = imopen(img, se);
% Fermeture : dilatation suivie d'érosion
img_fermeture = imclose(img, se);
figure;
subplot(1,3,1); imshow(img); title('Image originale');
subplot(1,3,2); imshow(img_ouverture); title('Ouverture');
subplot(1,3,3); imshow(img_fermeture); title('Fermeture');
```

Détection nouveaux d'objets :

Pour entraîner un modèle de détection d'objets dans MATLAB en utilisant l'Image Labeler App pour créer vos données annotées (labels), mais malheureusement notre pc ne contient pas une carte graphique. À cause de ça on obtient ce message.

```
>> imageLabeler("C:\Users\akram\OneDrive\Documentos\MATLAB\newLabels.mat")
Warning: While loading an object of class 'groundTruth':
Too many input arguments.
```

SSD :

Pour le SSD le même problème après fait la majorité des codes

Codes utilisés :

```
imds = imageDatastore("C:\Users\akram\OneDrive\Pièces jointes\Imágenes\Camera
Roll\New folder");
% Étape 2 : Données de boîtes englobantes et étiquettes (exemple avec une image)
bbox = [100 100 50 60]; % [x y largeur hauteur]
labels = categorical("person"); % étiquette
% Étape 3 : Créer une table contenant les boîtes englobantes
bboxTable = table({bbox}, 'VariableNames', {'person'});
% Étape 4 : Créer le boxLabelDatastore
blds = boxLabelDatastore(bboxTable);
% Étape 5 : Combiner les deux
trainingData = combine(imds, blds);
inputSize = [300 300 3]; % taille standard SSD
classes = {'person'}; % ou {'car','dog','cat'} etc.
baseNetwork = resnet50; % ou resnet18, mobilenetv2, etc.
featureLayer = 'activation_40_relu'; % dépend du réseau
numClasses = numel(classes); % nombre de classes = par ex. 1
inputSize = [300 300 3];
numClasses = 2;
baseNetwork = 'resnet50'; % ou 'mobilenetv2', 'resnet18', etc.
featureLayer = 'activation_40_relu'; % dépend du réseau !
net = resnet50;
analyzeNetwork(net)
lgraph = ssdLayers(inputSize, numClasses, baseNetwork, 'FeatureLayer', featureLayer);
```

Bibliographie

[1] « Niveaux de gris »

Lien : <https://retines.fr/blog/niveaux-de-gris/>

[2] Gonzalez, R.C., & Woods, R.E. (2018). Digital Image Processing (4th ed.). Pearson Education.

[3] MATLAB HELP « CONTOUR »

Lien : <https://www.mathworks.com/help/images/edge-detection.html>

[4] Mohammed Khamadja et Said Benierbah « Cour de Traitement d'images ». Université des frères Mentouri, Constantine 1

[5] Oussalem iman et SIBACHIR Abdenour « Détection et Reconnaissance d'objet avec MATLAB ». Mémoire de Master Université D'Oran 2 Mohamed Ben Ahmed Algerie, 2024.

[6] h. Guesmia, LALAOUI Lahouaoui « Algorithme de segmentation par chan-vese pour les différentes modalités des images » Université Mohamed Boudiaf, Msila, 2019.

[7] Laurent Caraffa, Jean Philippe Tarel, Pierre Charbonnier « The Guided Bilateral Filter: When the Joint/Cross Bilateral Filter Becomes Robust».

[8] KHWAB KALRA « YOLO », En 2023.

Lien : <https://medium.com/%40khwabkalra1/yolo-you-only-look-onc-523b01ec4f4d>

[9] Moulai Warda et Sari Assia. « Restauration d'image par filtre adaptatif stochastique en utilisant l'analyse multi-résolution ». Mémoire de Master, Université d'Oran 2 Mohamed Ben Ahmed, Oran, 2020.

[10] MERSAOUI Imane & FOUIRI Mohamed El Amine. « Etude sur l'imagerie médicale : prétraitement, segmentation et amélioration des exploitations ». Mémoire de Master, Université Saad Dahlab de Blida, Algerie. 2019.

[11] « Fuzzy Clustering – ML ». En 2025.

Lien : <https://www.geeksforgeeks.org/machine-learning/ml-fuzzy-clustering/>

[12] Bourahla Houda, Rabhi Chaima. « Matlab pour traitement de l'image fondement et applications ». Mémoire de Master, Université Mohamed Boudiaf De M'sila, Algerie. 2017.

[13] A. K. Jain « Fundamentals of Digital Image Processing » en 1989.

[14] LAKEHAL Elkhamssa « Classification et Reconnaissance d'Objets en Utilisant les Points d'Intérêt » Mémoire présenté en vue de l'obtention du diplôme de : Magister en Informatique, University de El Hadj Lakhdar, BATNA, 2009.

Bibliographie

[15] Nico Klingler « YOLOv4: A Fast and Efficient Object Detection Model », en 2024.

Lien : <https://viso.ai/deep-learning/yolov4/>

[16] « Le guide complet de la détection d'objets : Une introduction à la détection », 2024.

Lien : <https://visionplatform.ai/fr/le-guide-complet-de-la-detection-dobjets-une-introduction-a-la-detection-en-2024/>

[17] « Algorithmique et traitement de l'image » lycée académie de LYON. Lien :

<https://portail.lyc-la-martiniere-diderot.ac-lyon.fr/srv20/html/imageHTML/pages/page8.html>

[18] Khenfer Ismail, Bettayeb Elhadj Ali, Salhi Dhia El-Hak « Filtrage Adaptatif 2-D pour la Restauration d'images perturbées par du Bruit Impulsionnel » Université Kasdi Merbah, Ouargla, 2020.

[19] Faiz Ullah, Kamlesh Kumar, Tariq Rahim, Jawad Khan & Younhyun Jung « A new hybrid image denoising algorithm using adaptive and modified decision-based filters for enhanced image quality ». Gachon University, Sindh Madressatul Islam University, Kingston University. 2025

[20] Pierre Bonnet, « Filtrage médian », sur Lagis à l'Université de Lille, 2009.

[21] NACEREDDINE NAFAA & Mr GUESSOUM ABDERREZAK « Filtre Médian et ses Versions Améliorées pour la Réduction du Bruit Impulsionnel dans Les Images Radiographiques de Soudure » Mémoire de Master, Université de saad dahleb, blida, 2019.

[22] Lien : <https://www.scribd.com/document/592563191/Les-differents-types-de-bruit>

[23] Gaëtan Berthouly « Bruit en Photo : Causes, Solutions et Techniques pour des Images Nettes ». En 2025.

Lien : <https://photomaniac.fr/tout-savoir-sur-le-bruit-en-photo-lexplication-complete>

[24] Laboratoire d'Acoustique de l'Université du Maine (LAUM), UMR CNRS 6613, Avenue Olivier Messiaen, 72085 Le Mans Cedex 9, France

[25] Isabelle Bellin « Histoire du traitement d'images ». En 2004.

Lien : <https://interstices.info/histoire-du-traitement-dimages/>

[26] « L'histoire de la reconnaissance d'images ». En 2021.

Lien : <https://praedictia.com/page/la-reconnaissance-dimages/lhistoire-de-la-reconnaissance-dimages.html>

Ce site de formation a été conçu par la SOFAD grâce au programme NovaScience – Soutien aux initiatives de formation en intelligence artificielle (IA) du ministère du Développement de l'Économie et de l'Innovation.

[27] Benjamin Perret. « Convolution ». Documentation Traitement et analyse d'images, 2017.

Lien : <https://perso.esiee.fr/~perretb/I5FM/TAI/convolution/index.html>

Bibliographie

[28] Mekideche Mohammed « Traitement d'Images par des Algorithmes Basés sur le Calcul Fractionnaire » Mmoire Doctorat Faculté de Technologie Département de Génie Electrique, Université 20 Août 1955, Skikda, 2018.

[29] Mariee Hillion « 10 types de format d'image et leur utilisation ». En 2025.

Lien : <https://blog.hubspot.fr/website/differents-types-fichiers-image>

[30] « L'image numérique »

Lien : <http://www.map.toulouse.archi.fr/works/panoformation/imagenum/imagenum.htm>

[30] Melle MEDJAHED Fatiha « Détection et Suivi d'Objets en Mouvement Dans Une Séquence d'Images » Mémoire en vue de l'obtention du diplôme de Magister Université d'Oran U. S. T. O. Mohamed Boudiaf, 2012.

[31] ABDICHE Ibrahim & NEGHLI Yousouf « Extraction de la région d'intérêt à partir d'images par les techniques de segmentation ». Mémoire de MASTER. UNIVERSITE IBN KHALDOUN, TIARET. 2022

[32] Lachaal Houda & Houmeur Hanane « Utilisation de constances des couleurs pour la segmentation des images » Mémoire de Fin d'études Master. Université de 8 Mai 1945, Guelma. 2017

[33] Mohamed SANDELI « Traitement d'images par des approches bio-inspirées Application à la segmentation d'images ». Mémoire En vue de l'obtention du diplôme de Magister en informatique. Université Constantine 2. Constantine. 2014

[34] KERKACHA Messaouda « LA SEGMENTATION D'IMAGES » Projet de fin d'étude présenté en vue de l'obtention du diplôme de LICENCE. Université de Ghardaïa, Ghardaïa. 2014

[35] Rawdha Bahri & Nour elhouda Hadjadj « Segmentation d'Image par Classification (k-Means) » Mémoire MASTER. Université KASDI MERBAH, OUARGLA. 2020

[36] DICKO ALLASANE & DOUIDI ASSIA « Mise en place d'un système d'indexation et de recherche d'image par le contenu » Mémoire Présenté pour l'obtention du diplôme de MASTER Université Aboubakr Belkaïd, Tlemcen. 2022

[37] Pratt, W. K. « Digital Image Processing: PIKS Scientific Inside (4th ed.) ». Wiley. 2007

[38] McRobbie, D. W., et al. (2006). « MRI: From Picture to Proton (2nd ed.) ». Cambridge Université. 2006

[39] Govender, M., Chetty, K., & Bulcock, H. « A review of hyperspectral remote sensing and its application in vegetation and water resource studies », 2017.

[40] IONOS SARL « Qu'est-ce qu'un pixel ? ». En 2025.

Lien : <https://www.ionos.fr/digitalguide/sites-internet/web-design/quest-ce-quun-pixel/>

Bibliographie

[41] Tinku Acharya & Ajoy K. Ray “Image Processing Principles and Applications” A JOHN WILEY & SONS, MC, PUBLICATION “

[42] ZERROUKI Abdel Aziz & BOUDIB Mohamed Taki ALLAH « Etude et réalisation d'un système de reconnaissance de faces humaines par la méthode des faces propres » Mémoire de Master Université SAAD DAHLAB de BLIDA, 2016.

[43] NEDJAR Yasmina & MOUSSI Imane Sujet « Application des méthodes numériques de traitement d'image sous Android » mémoire présenté pour l'obtention du diplôme de MASTER en Télécommunications Université Aboubakr Belkaïd de Tlemcen, 2018.

[44] Mme. Azouza Naouel « Segmentation par classification floue :application à l'imagerie par résonance magnétique » MEMOIRE Présenté pour l'obtention du diplôme de MASTER. UNIVERSITE AKLI MOAND OULHADJE, BOUIRA. 2018

[45] Mehdi Felhi, Salvatore Tabbone, Maria V. Ortiz Segovia « Approche hybride de segmentation de page à base d'un descripteur de traits ». Université de Lorraine, Nancy, France. Océ Print Logic Technologies- Canon Group, Créteil, France. En 2024

[46] « Qu'est-ce que la segmentation d'images ? ».

Lien : <https://www.ibm.com/fr-fr/think/topics/image-segmentation>

[47] Wiharto, Dr, «The Comparison of Clustering Algorithms K-Means and Fuzzy C-Means for Segmentation Retinal Blood Vessel ». Department of Informatics, Universitas Sebelas Maret, Jl. Ir. Sutami, No. 36A, Kentingan, Surakarta, Indonesia.

Lien : <https://pmc.ncbi.nlm.nih.gov/articles/PMC7085333/>

[48] « Qu'est-ce que c'est : le seuillage ». En 2025

Lien : <https://fr.statisticseasily.com/glossaire/qu'est-ce-que-le-seuillage-dans-le-traitement-d'image/>

[49] « Seuillage d'image ». Lien : https://fr.wikipedia.org/wiki/Seuillage_d'image

[50] Zouaoui HAKIMA « Clustering par fusion floue de données appliqué à la segmentation d'images IRM cérébrales », Mémoire de magister en informatique à l'université Mohammed Bougara de Boumerdes, 2008.

[51] « Traitement d'images (partie 5 : Transformations morphologiques) » A propos de Benoit Cayla. 2024.

Lien : <https://datacorner.fr/image-processing-5/>

[52] Aissa Boulmerka « Chapitre 06 Traitement morphologique de l'image » université de mila. 2020-2021

[53] « Chap. 7 Traitement d'Images Binaires Par Morphologie Mathématique ». Réalisé par clément valley en Feb 18, 2023.

[54] « Morphologie Mathématique ». Réalisé par Benjamin Perret en 2017

Bibliographie

Lien : <https://perso.esiee.fr/~perretb/I5FM/TAI/morpho/index.html>

[55] « La segmentation d'images : clé de l'intelligence artificielle visuelle ? » Ecrit par

Daniella. 2024. Lien : <https://www.innovatiana.com/fr/post/image-segmentation-in-ai>

[56] « The Complete Guide to Image Preprocessing Techniques in Python » Par Maahi Patel en 2023. Lien : <https://medium.com/%40maahip1304/the-complete-guide-to-image-preprocessing-techniques-in-python-dca30804550c>

[57] « Extraction de caractéristiques dans l'apprentissage automatique : Un guide complet » écrit par Rajesh Kumar en 2025

Lien : <https://www.datacamp.com/fr/tutorial/feature-extraction-machine-learning>

[58] « R-CNN - Region-Based Convolutional Neural Networks » en 2025.

Lien : <https://www.geeksforgeeks.org/machine-learning/r-cnn-region-based-cnns/>

[59] « What is R-CNN ? ». Écrit par Petru P. en 2023.

Lien : <https://blog.roboflow.com/what-is-r-cnn/>

[60] « Vision par ordinateur Amazon » en 2024

Lien : <https://aws.amazon.com/fr/what-is/computer-vision/>

[61] Eddie Forson « Understanding SSD MultiBox – Real-Time Object Detection In Deep Learning ». En 2017.

Lien : <https://towardsdatascience.com/understanding-ssd-multibox-real-time-object-detection-in-deep-learning-495ef744fab/>

[62] « Getting Started with SSD Multibox Detection » MATLAB HELP.

Lien : <https://www.mathworks.com/help/vision/ug/getting-started-with-ssd.html>

[63] « Getting Started with YOLO v2 » MATLAB HELP.

Lien : <https://www.mathworks.com/help/vision/ug/getting-started-with-yolo-v2.html>

[64] Sebaa Brahim & Belmesmar Djamila « Traitement d'image et morphologie mathématique » Mémoire MASTER. Université KASDI MERBAH, OUARGLA. 2016

[65] « Otsu's method » Wikipedia. 2025

Lien : https://en.wikipedia.org/wiki/Otsu%27s_method?utm_source=chatgpt.com

[66] Patrick Hébert & Denis Laurendeau « Cour Traitement des images (partie 1 : pré-traitement) ». Université LAVAL, Canada. 2017. Lien :

<https://wcours.gel.ulaval.ca/2017/a/GIF4100/default/5notes/A2017TraitementImagesPartie1PageWeb.pdf>

[67] « Object Detection Using Faster R-CNN Deep Learning ». MATLAB HELP

Bibliographie

Lien : <https://www.mathworks.com/help/vision/ug/object-detection-using-faster-r-cnn-deep-learning.html>

[68] Housseem Eddine « La détection d'objet avec OpenCV et deep Learning » MÉMOIRE DE MASTER. Université Mohamed Khider de Biskra, 2020.

[69] Saurav Kumar, Drishti Yadav, Himanshu Gupta, Om Prakash Verma, Irshad Ahmad Ansari et Chang Wook Ahn « A Novel YOLOv3 Algorithm-Based Deep Learning Approach for Waste Segregation: Towards Smart Waste Management ». En 2020.

Lien : <https://www.mdp.com/2079-9292/10/1/14>

[70] Abirami Vina « What is Mask R-CNN and how does it work? ». En 2025.

Lien : <https://www.ultralytics.com/blog/what-is-mask-r-cnn-and-how-does-it-work>

[71] « Le bruit gaussien ». Lien :

https://media4.obspm.fr/public/ressources_lu/pages_analyser/bruit-gaussien_impression.html