République Algérienne Démocratique Et Populaire
Université Mohamed Boudiaf De M'sila
Faculté Des Mathématiques Et De L'informatique
Département De Mathématiques

MEMOIRE DE FIN D'ETUDE

Présenté pour l'obtention du Diplôme de **MASTER**

Domaine : Mathématiques et Informatique

Filière : Mathématiques

Option : Equation aux Dérivées Partielles applications

Par

-Bourahla Houda - Rabhi Chaima

Sujet

Matlab pour traitement de l'image fondement et applications

Date de soutenance :03/05/2017

: Devant le jury

Président	Prof. Univ de M'sila	Mr.Merzougui .A
Rapporteur	Prof. Univ de M'sila	Mr.Ben Hamidouche. N
Examinateu -se	Prof. Univ de M'sila	Mm. Bounab .N

بِسْمِ اللَّهِ الرَحمَنِ الرَحِيمْ

"وَقُلِ اعْمَلُواْ فَسَيَرَى اللّهُ عَمَلَكُمْ وَرَسُولُهُ وَالْمُؤْمِنُونَ وَسَتُرَدُّونَ إِلَى عَالِمِ الْغَيْبِ وَالشَّهَادَةِ فَيُنَبِّئُكُم بِمَا كُنتُمْ تَعْمَلُونَ" [التوبة: 105]

ويقول العماد الأصفهاني

(إني رأيت أنه لا يكتب أحد كتابا في يومه إلا قال فيه عنه : لو غير هذا لكان أحسن ، ولو زيد لكان يستحسن ، ولو قدم هذا لكان أفضل ، ولو ترك هذا لكان أجمل ، وهذا من أعظم العبر ، وهو دليل على استيلاء النقص على جملة البشر).

Remerciements

Nous tenons tout d'abord à remercier ALLAH le tout puissant de nous avoir donné le courage et la patience pour mener à bien ce modeste travail,
qu'il soit béni et glorifié.
Nous adressons également nos remerciements à Ms BENHAMIDOUCHE N, pour avoir accepté d'être notre encadreur.
Nous le remercions aussi pour ses conseils, ses corrections et ses orientations.
Nous tenons aussi à exprimer mes remerciements les plus respectueux
à messieurs les membres du jury.

Merci

Table des matières

Introduction.

Le traitement d'images est un domaine très vaste qui a connu, et qui connaît encore, un développement important depuis quelques dizaines d'années. On désigne par traitement d'images numériques l'ensemble des techniques permettant de modifier une image numérique afin d'améliorer ou d'en extraire des informations. De ce fait, le traitement d'images est l'ensemble des méthodes et techniques opérant sur celles-ci, dans le but de rendre cette opération possible, plus simple, plus efficace et plus agréable, d'améliorer l'aspect visuel de l'image et d'en extraire des informations jugées pertinentes.

Matlab est un logiciel qui est pertinent dans le domaine du traitement d'image, il est parmis les outils les plus utilisés dans ce domaine. MATLAB est une abréviation de Matrix LABoratory, il a été conçu par Cleve Moler à la fin des années 1970, c'est un environnement puissant, complet et facile à utiliser. Il apporte aux ingénieurs, chercheurs et à tout scientifique un système interactif intégrant calcul numérique et visualisation. C'est un environnement performant, ouvert et programmable qui permet de remarquables gains de productivité et de créativité. MATLAB comprend un ensemble d'outils spécifiques à des domaines, en particulier en traitement de l'image, appelés Toolboxes (ou Boîtes à Outils).

Le but de ce mémoire est d'illustrer en détail les commandes et instructions contenues dans la boite à outils dédiée au traitement de l'image, notamment les manipulation des images, le filtrage linéaire et non linéaire, la restauration d'image et enfin la détection des contours.

Ce mémoire est composé de trois chapitres:

Le premier chapitre comprend des défintions générales sur l'image, et sur le prétraitemtent de l'image en particulier, ainsi que quelques propriétés générales liées à l'image.

Dans le chapitre deux, on présente les différentes technique du traitement de l'image sous matlab, notamment , le filtrage, le bruitage, le débruitage, la restauration d'image, et le principe de la déconvolution.

Cette étude est illustrée par des programmes et instruction compliées sur matlab sur des images qu'on a choisi.

Enfin dans le dernier chapitre, une présentation des differentes méthodes appliquées dans la détection des contours d'image, notamment celle de Canny ,de roberts,de perwitt,etc.... Pour toutes ces méthodes des exemples d'illustrations sont présentés.

Chapitre 1

Définitions et propriètés générales

1.1 Image mathématique

Définition 1.1.1 *Une image est plutôt difficile à décrire d'une façon générale. En traitement d'image, la majorité du temps, on considère qu'il s'agit d'une fonction mathématique de $R*R$ dans R où le couplet d'entrée est considéré comme une position spatiale, le singleton de sortie comme l'intensité (couleur ou niveaux de gris) du phénomène physique. Il arrive cependant que l'image soit dite "3D" donc la fonction est de$R*R*R$ dans R. Les images couleurs peuvent être représentées soit par trois images représentant les trois couleurs fondamentales, soit par une image de $R*R$ dans $R*R*R$. Soit Ω un ouvert borné de $\mathbb{R}^2$, une image est définie comme une fonction.*

$$f : \Omega \longrightarrow \mathbb{R} \textit{ pour une image niveau de gris}$$

$$f : \Omega \longrightarrow \mathbb{R}^3 \textit{pour une image couleur}$$

1.2 Image numérique

Définition 1.2.1 *L'image numérique est l'image dont la surface est divisée en éléments de tailles fixes appelés cellules ou pixels, ayant chacun comme caractéristique un niveau de gris ou de couleurs prélevé à l'emplacement correspondant dans l'image réelle, ou calculé à partir d'une description interne de la scène à représenter.*

Définition 1.2.2 ***Le pixel:*** *Une image numérique est constituée d'un ensemble de points appelés pixels (abréviation de PICture Element) pour former une image. Le pixel représente ainsi le plus petit élément constitutif d'une image numérique. L'ensemble de ces pixels est contenu dans un tableau à deux dimensions constituant l'image. Autrement dit, une image est une matrice $M \times N$ de valeurs entières prises sur un intervalle borné $[0, Ng]$ où Ng est la valeur maximale du niveau de gris. p(i,j) est le niveau de gris du pixel de coordonnées ligne i et colonne j dans l'image. $p(i,j) \in [0, Ng]$. Les valeurs des niveaux de gris sont des entiers.*

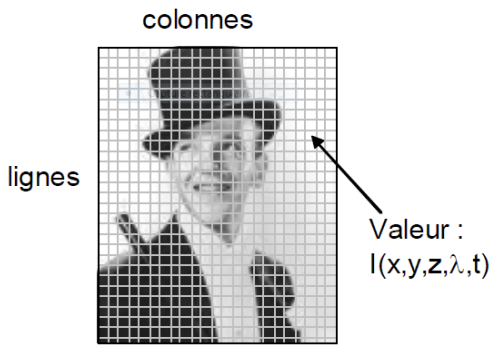

Figure 1: Representation du pixel

1.3 Les types d'image:

1. **Image binaire:**

Définition 1.3.1 *Une image binaire est une image $M \times N$ où chaque point peut prendre uniquement la valeur 0 ou 1. Les pixels sont noirs (0) ou blancs (1).*

Le niveau de gris est codé sur un bit . Dans ce cas, avec Ng=2 et la relation sur les niveaux de gris devient: $p(i,j) = 0$ *ou* $p(i,j) = 1$.(*voir* [*3*])

Figure 2: Une image binaire

2**. Image en niveaux de gris:**

Définition 1.3.2 *Le niveau de gris est la valeur de l'intensité lumineuse en un point. La couleur du pixel peut prendre des valeurs allant du noir au blanc en passant par un nombre fini de niveaux intermédiaires. Donc pour représenter les images à niveaux de gris, on peut attribuer à chaque pixel de l'image une valeur correspondant à la quantité de lumière renvoyée. Cette valeur peut être comprise par exemple entre 0 et 255. Chaque pixel n'est donc plus représenté par un bit, mais par un octet. Pour cela, il faut que le matériel utilisé pour afficher l'image soit capable de produire les différents niveaux de gris correspondant. Une image de niveaux de gris autorise un dégradé de gris entre le noir et le blanc. En général, on code le niveau de gris sur un octet (8 bits) soit 256 nuances de dégradé. L'expression de la valeur du niveau de gris avec Ng =256 devient:* $p(i,j) \in [0, 255]$.

Figure 3:Une image niveau de gris

3. **Image couleur:**

Définition 1.3.3 *Même s'il est parfois utile de pouvoir représenter des images en noir et blanc, les applications multimédias utilisent le plus souvent des images en couleurs. La représentation des couleurs s'effectue de la même manière que les images monochromes avec cependant quelques particularités. En effet, il faut tout d'abord choisir un modèle de représentation. On peut représenter les couleurs à l'aide de leurs composantes primaires. Une image couleur est la composition de trois (ou plus) images en niveaux de gris sur trois (ou plus) composantes. On définit donc trois plans de niveaux de gris, un rouge, un vert et un bleu. La couleur finale est obtenue par synthèse additive des ces trois (ou plus) composantes.*

On a les relations sur les niveaux de gris:$P_R(i,j) \in [0,255]$, $P_V(i,j) \in [0,255]$, $P_B(i,j) \in [0,255]$.

$$\text{Image} = \text{matrice de } M \times N \text{ élements, ou une fonction } f(x,y)$$

4-**Image indéxée:**

Définition 1.3.4 *La plupart des images en couleur n'ont qu'un petit sous-ensemble de plus de seize millions possible couleurs. Pour la commodité du stockage et de la gestion des fichiers, l'image comporte une carte de couleur associée ou une palette de couleurs, qui est simplement une liste de toutes les couleurs utilisées dans cette image. Chaque pixel a une*

valeur qui ne donne pas sa couleur (comme pour une image RVB), mais un index de la couleur dans la carte.(voir[3])

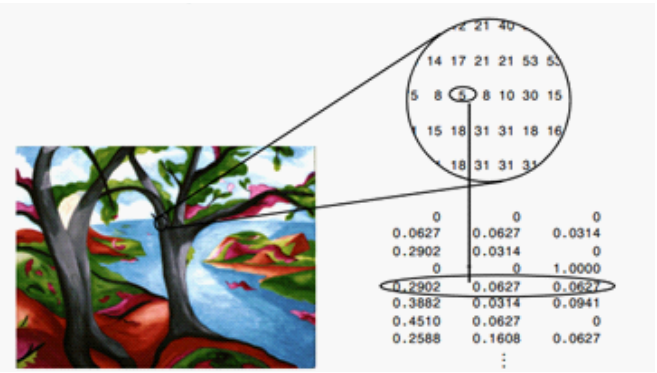

Figure 4 :Une image indéxée

1.4 Caractéristiques d'une image numérique

L'image est un ensemble structuré d'informations caractérisé par les paramètres suivants:

- Dimension :

C'est la taille de l'image. Cette dernière se présente sous forme de matrice dont les éléments sont des valeurs numériques représentatives des intensités lumineuses (pixels). Le nombre de lignes de cette matrice multiplié par le nombre de colonnes nous donne le nombre total de pixels dans une image.

- Résolution :

La résolution est le nombre de bits associés à chaque couleur primaire d'un pixel. Cette valeur reflète le nombre de couleurs ou de niveaux de gris d'une image : 32 bits = 1,07 milliards de couleurs 24 bits = 16,7 millions de couleurs (ou couleurs vraies) 16 bits = 65 536 couleurs 8 bits = 256 couleurs 1 bit = 2 couleurs (noir et blanc) .

- Bruit :

Un bruit dans une image est considéré comme un phénomène de brusque variation de l'intensité d'un pixel par rapport à ses voisins, il provient de l'éclairage des

dispositifs optiques et électroniques du capteur .

-Histogramme :

L'histogramme des niveaux de gris ou des couleurs d'une image est une fonction qui donne la fréquence d'apparition de chaque niveau de gris (couleur) dans l'image. Il permet de donner un grand nombre d'information sur la distribution des niveaux de gris (couleur) et de voir entre quelles bornes est repartie la majorité des niveaux de gris (couleur) dans le cas d'une image trop claire ou d'une image trop foncée.

Il peut être utilisé pour améliorer la qualité d'une image (Rehaussement d'image) en introduisant quelques modifications, pour pouvoir extraire les informations utiles de celle-ci.

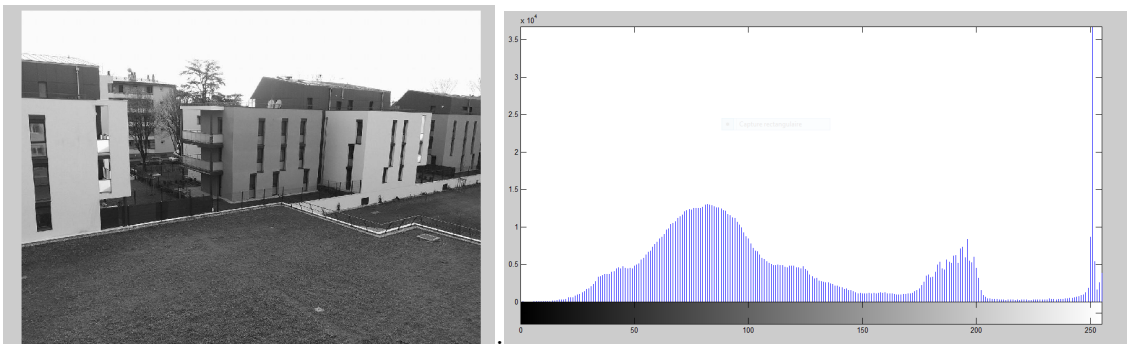

Figure 5: Une image niveau de gris et son histogramme

- Luminance:

Définition 1.4.1 *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, pour un observateur lointain, le mot luminance est substitué au mot brillance, qui correspond à l'éclat d'un objet. 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.

On peut calculer une image en niveaux de gris à partir d'une image couleur en moyennant les trois canaux. On calcule donc une valeur a qui s'appelle **la luminance** de la couleur.

$$a = \frac{r + v + b}{3}$$

- **Contraste:**

Définition 1.4.2 *C'est l'opposition marquée entre deux régions d'une image, plus précisément entre les régions sombres et les régions claires de cette image. Le contraste est défini en fonction des luminances de deux zones d'images. Si* $L1$ *et* $L2$ *sont les degrés de luminosité respectivement de deux zones voisines* $A1$ *et* $A2$ *d'une image , le contraste* C *est défini par le rapport :*

$$\mathbf{c} = \frac{l1 - l2}{l1 + l2}$$

1.5 Les formats d'images

Les formats de fichiers d'image les plus courants, les plus importants pour les caméras, l'impression, la numérisation et l'utilisation d'Internet, sont JPG, TIF, PNG et GIF.

- Le **JPG** est utilisé lorsque la taille du fichier est plus importante que la qualité maximale de l'image (pages Web, courrier électronique, cartes mémoire, etc.)
- **TIF** est sans perte (y compris l'option de compression LZW), qui est considéré comme le format de la plus haute qualité pour les travaux commerciaux.
- **GIF** a été conçu par CompuServe dans les premiers jours de la vidéo 8 bits par ordinateur, avant JPG, pour l'affichage vidéo à la vitesse de modem téléphonique.
- **PNG** peut remplacer GIF aujourd'hui (les navigateurs Web montrent les deux), et PNG offre également de nombreuses options de TIF (indexées ou RVB, 1 à 48 bits, etc.).(voir[1])

1.6 Principe et domaine d'application

Traitement d'images

Définition 1.6.1 *:Le traitement d'images est une discipline de l'informatique et des mathématiques appliquées qui étudie les images numériques et leurs transformations, dans le but d'améliorer leur qualité ou d'en extraire de l'information. Il s'agit d'un sous-ensemble du traitement du signal dédié aux images et aux la vidéos . Dans le contexte de la vision artificielle, le traitement d'images se place après les étapes d'acquisition et de numérisation, assurant les transformations d'images et la partie de calcul permettant d'aller vers une interprétation des images traitées. Cette phase d'interprétation est d'ailleurs de plus en plus intégrée dans le traitement d'images, en faisant appel notamment à l'intelligence artificielle pour manipuler des connaissances, principalement sur les informations dont on dispose à propos de ce que représentent les images traitées (connaissance du« domaine »). La compréhension du traitement d'images commence par la compréhension de ce qu'est une image. Le mode et les conditions d'acquisition et de numérisation des images traitées conditionnent largement les opérations qu'il faudra réaliser pour extraire de l'information. En effet, de nombreux paramètres entrent en compte, les principaux étant : la résolution d'acquisition et le mode de codage utilisé lors de la numérisation, qui déterminent le degré de précision des éventuelles mesures de dimensions.*

Le domaine d'application lié au traitement de l'image est trés varié:

- la compression d'imge video.

- L'imagerie satellitaire.

- L'imagerie médicale.

- La vission artificielle.

- les nouvelles technologie del'image, etc...

Histoire de traitement d'image:

Le traitement de l'image numérique a vu son utilisation à New York, aux États-Unis au début des années 1920. Il a d'abord été utilisé pour améliorer les images de journaux envoyées par câble sous-marin entre Londres et New York. Cependant, l'utilisation du traitement d'image numérique n'est pas devenue populaire avant les années 1960, lorsque

les ordinateurs numériques de troisième génération ont commencé à offrir la vitesse et les capacités de stockage requises pour la mise en œuvre pratique des algorithmes de traitement d'image. Le domaine du traitement de l'image numérique a connu une croissance considérable depuis les années 60. L'utilisation du traitement de l'image est sans fin ayant fait l'objet d'une étude et d'une recherche interdisciplinaires dans des domaines tels que l'ingénierie, l'informatique, la médecine et d'autres domaines tels que l'application de la loi et le transport .

Histoire de MATLAB:

MATLAB est à la fois un langage de programmation et un environnement de développement développé et commercialisé par la société américaine MathWorks. MATLAB est utilisé dans les domaines de l'éducation, de la recherche et de l'industrie pour le calcul numérique mais aussi dans les phases de développement de projets.

Le nom **MATLAB** est la contraction du terme anglais matrix laboratory. Le langage MATLAB a été conçu par Cleve Moler à la fin des années 1970 à partir des bibliothèques Fortran, LINPACK et EISPACK. Alors professeur de mathématiques à l'Université du NouveauMexique, il souhaitait permettre à ses étudiants de pouvoir utiliser ces deux bibliothèques sans connaitre le Fortran. Cleve Moler l'utilisa ensuite pour des cours donnés à l'université Stanford où il reçut un accueil mitigé de la part des étudiants en mathématiques (habitués au Fortran). Par contre, les étudiants en technologie, en particulier en traitement du signal, furent beaucoup plus intéressés. Un ingénieur, Jack Little en comprend rapidement les capacités et entreprend avec un collègue, Steve Bangert, de le recoder en langage C. Jack Little, Cleve Moler et Steve Bangert créèrent la société The MathWorks en 1984 afin de commercialiser la version 1.0 de MATLAB. MATLAB a ensuite évolué, en se dotant de nombreuses boites à outils (Toolbox) et en incluant les possibilités données par d'autres langages de programmation comme C++ ou Java.([2])

1.7 Matlab et traitements d'images

Une image Matlab est une matrice bidimensionnelle de valeurs entières ou réelles. Les principales fonctions de traitement d'images sous Matlab se trouvent dans la boîte à outils

(toolbox) image processing (traitement d'images). L'aide sur cette boîte à outils est obtenue en tapant **help** images en ligne de commande de Matlab. Ensuite, l'aide sur une commande particulière est obtenue en tapant help suivi du nom de la commande, par exemple **help edge**.

Matlab est un logiciel de calcul scientifique permettant de développer des solutions à des problèmes techniques. Il permet de réaliser du calcul numérique et tracer des graphiques pour visualiser et analyser les données. Il dispose d'un langage et d'un environnement de programmation interactifs ainsi que d'outils pour concevoir des interfaces utilisateur-graphiques. Matlab est associé à des boites à outils appelées toolbox permettant d'accéder à des fonctions spécifiques à un domaine d'application comme le traitement d'images par exemple. Les TD et TP de vision réalisés avec Matlab nécessitent ainsi les toolbox **image acquisition** et **image processing**

Après avoir rappelé quelques généralités sur Matlab, nous allons découvrir les fonctions de base permettant d'acquérir, d'afficher et de sauvegarder une image. Les principaux outils de traitements d'images seront ensuite abordés et appliqués à des problématiques concrètes, telles que la compression, la restauration, la segmentation, ...

1.8 Matlab et Prétraitement d'image:

Pour manipuler une image, on travaille sur un tableau d'entiers qui contient les composantes de chaque pixel. Les traitements s'appliquent toujours aux images en niveau gris et parfois aussi sur des images couleur. Nous allons distinguer plusieurs types d'opérations sur l'image

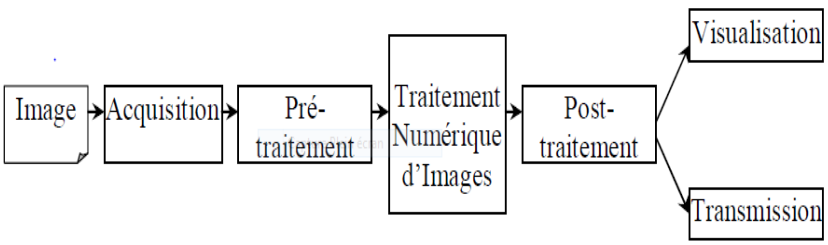

Figure 6: Schéma d'un système de traitement d'images

Acquisition d'une image:

Acquisition d'image: c'est la première étape ou le processus des étapes fondamentales du traitement d'image numérique. L'acquisition d'image pourrait être aussi simple que d'avoir une image qui est déjà sous forme numérique. Généralement, l'étape d'acquisition d'image implique un prétraitement, comme la mise à l'échelle,

Le chargement en mémoire d'une image se fait avec la fonction **imread**. Par exemple, la fonction suivante permet de lire une image et de placer son contenu dans une variable de type matrice, voila par exemple l'image qu'on a prise:

I=imread('houda.tif')

Cette variable est alors visible dans le Workspace (espace des variables) de Matlab. La fonction qui permet d'afficher toutes les informations relatives aux variables en mémoire et la fonction **imfinfo** affiche les informations relatives à un fichier image.

L'affichage de l'image (ou de la variable) est réalisé par la fonction imshow. Ainsi les fonctions suivantes ouvrent une nouvelle fenètre pour y afficher l'image I.

figure;imshow(I);

1.9 Manipulation des images sous Matlab

La somme de deux images:

Z = imadd (X, Y)

ajoute chaque élément dans le tableau X avec l'élément correspondant dans le tableau Y et renvoie la somme dans l'élément correspondant du tableau de sortie Z. X et Y sont des tableaux numériques réels non séparés de même taille et classe , Ou Y est un double scalaire. Z a la même taille et la même classe que X, sauf si X est logique, auquel cas Z est double.

Si X et Y sont des tableaux entiers, les éléments de la sortie qui dépassent la portée du type entier sont tronqués et les valeurs fractionnaires sont arrondies.

Exemple 1.9.1 *1-Lire deux images uint8 en niveaux de gris dans l'espace de travail*

I = imread('rice.png');

J = imread('cameraman.tif');

2-Ajouter les images. Spécifiez la sortie comme type uint16 pour éviter de tronquer le résultat.

K = imadd(I,J,'uint16');

3-Afficher le résultat.

imshow(K,[ ])

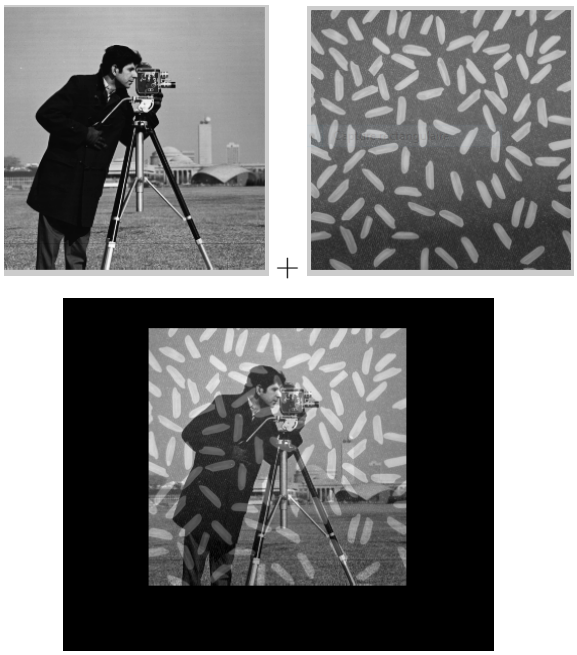

Figure 7 : La somme de deux images

On peut ajouter un contraste dans une image , comme suite

I = imread('rice.png');

J = imadd(I,50);

subplot(1,2,1);imshow(I);

subplot(1,2,2);imshow(J);

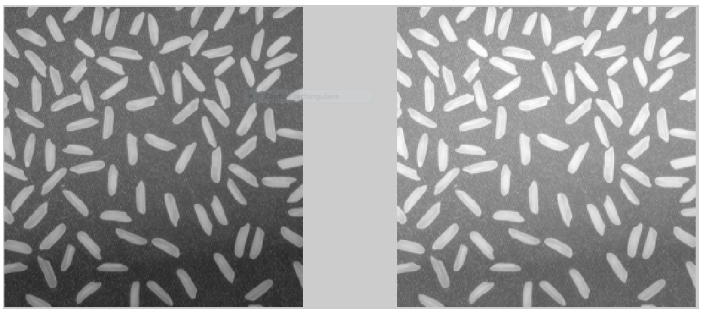

Figure 8 :Le resultat avant et aprés le contraste

Fonctions de matlab

Certaines fonctions ou certains outils de Matlab permettent des manipulations interactive sur une image contenue dans une figure ou non :

- imageinfo : retourne les information de l'image dans la figure ou d'un fichier image,
- zoom : zoom sur une zone de l'image de la figure,
- Imnoise : Ajouter du bruit à l'image,
- improfile : affiche le profil d'une ligne sélectionnée,
- impixel : retourne les valeurs des pixels sélectionnés,
- impixelinfo : affiche la position et les valeurs d'un pixel pointé avec la souris,
- impixelregion : affiche les valeurs des pixels dans une région sélectionnée avec la souris,
- imdistline : affiche la distance entre deux pixels sélectionnés,
- imdisplayrange : affiche l'intervalle des valeurs des pixels de l'image,
- imcontrast : réajuste une image,
- Imhist : Histogramme des données d'image.
- Histeq : Améliorer le contraste en utilisant l'égalisation des histogrammes

- imresize : ré-échantillonnage de l'image (homothétie),
- imrotate : rotation de l'image.
- imadjust : recadrage de la dynamique selon une correction gamma,

les types de conversion d'image

- gray2ind->Convertir l'image en niveaux de gris ou l'image binaire en image indexée
- ind2gray->Convertir l'image indexée en image en niveaux de gris
- mat2gray->Convertir la matrice en image en niveaux de gris
- rgb2gray->Convertir l'image RVB ou la carte des couleurs en niveaux de gris
- ind2rgb->Convertir l'image indexée en image RVB
- label2rgb->Convertir la matrice en image RVB
- demosaic->Convertir l'image codée du modèle Bayer en image couleur vraie
- im2bw->Convertir l'image en image binaire, en fonction du seuil
- im2double->Convertir l'image en double précision
- im2int16->Convertir l'image en entiers signés 16 bits(voir[7])

Exemple 1.9.2 *cet exemple montre la rotation d' image :*

Figure 9: La rotation d'une image

Exemple 1.9.3 *Cet exemple transforme une image en niveau de gris .*

Y = imread('houda.jpg');

X = rgb2gray(Y);

subplot(1,2,1);imshow(Y)

subplot(1,2,2);imshow(X)

Figure 10: Conversion d'une image couleur a en niveau de gris

Exemple 1.9.4 *Créer l'image complémentaire d'une intensité*

I = imread('cameraman.tif');

J = imcomplement(I);

imshowpair(I,J,'montage')

Figure 11 :Image et son complémentire

Exemple 1.9.5 *I = imread('pout.tif');*

J = imadjust(I);

K = imadjust(I,[0.3 0.7],[]);

subplot(1,3,1);imshow(I);

subplot(1,3,2);imshow(J);

subplot(1,3,3);imshow(K)

Figure 12 : Image ajustée

Exemple 1.9.6 *Améliorer le contraste d'une image d'intensité à l'aide de l'égalisation d'histogramme.*

H=imread('E:\houda.jpg');

I=rgb2gray (H);

J = histeq(I);

subplot(1,2,1);imshow(I);

subplot(1,2,2);imshow(J);

Figure 13: Amélioration du contraste

Chapitre 2

La restauration d'image sous MATLAB

2.1 Introduction

La restauration d'images tend à améliorer la qualité (souvent visuelle) d'une image , par exemple une photographie dont le négatif aurait reçu une poussière conduira à un défaut sur l'image . Un mouvement lors de l'acquisition d'une photographie pourra conduire à une image floue. Une sous-illumination conduira à des couleurs erronnées. Certains défauts peuvent étre éliminés ou réduits. Mais il ne faut pas non plus attendre que la restauration d'images permettent de restaurer tout et n'importe quoi. Meilleure sera la prise d'image, meilleure sera l'image ensuite. Son but est d'obtenir une image qui soit la plus proche possible de l'image idéale qui aurait été obtenue si le système d'acquisition était parfait.

Dans la restauration d'image on retrouve, le filtrage qui est une technique pour modifier ou améliorer une image, par exemple, on peut filtrer une image pour souligner certaines fonctionnalités ou supprimer d'autres fonctions, il existe deux types de filtrage linéaire et non linéaire. Le principe de filtrage linéaire se base sur la convolution, par contre la restauration d'image dans le cas par exemple de rétablir des documents de détérioris ce-ci est determinée par une opération de déconvolution .

2.2 Filtrage d'une image

Définition 2.2.1 *Le filtrage dans le traitement d'image est un processus qui permet de nettoyer les apparences et permet de mettre en surbrillance sélective des informations spécifiques. Un certain nombre de techniques sont disponibles et les meilleures options peuvent dépendre de l'image et comment elles seront utilisées. Le traitement d'image analogique et numérique peut nécessiter un filtrage pour produire un résultat final utilisable et attrayant. Cela peut être une partie courante du processus d'édition utilisé pour préparer les images à distribuer.(voir[2])*

Pourquoi filtrer

Le filtrage d'une image est effectué pour assurer certaines fonctionnalités qui incluent :

- Eliminer ou réduire le bruit dans une image.
- Détecter les bords d'une image.
- Convolution entre une image et un filtre.

Les filtres peuvent être séparés en deux catégories, filtres linéaires et non linéaires.

2.3 Filtrage linéaire

Le filtrage linéaire d'une image s'effectue par une opération appelée convolution. Soit I une image numérique , soit h une fonction de $[x_1, x_2] \times [y_1, y_2]$ à valeurs réelles,

la convolution de I par h est définie par :

$$(I * h)[x, y] = \sum_{i=x_1}^{x_2} \sum_{j=y_2}^{y_1} h(i, j)[x - i, y - j]$$

tels que h est un noyaux de convolution.

Par exemple, supposons que l'image soit :

$$A = \begin{matrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{matrix}$$

Et le noyau de convolution est :

$$h = \begin{matrix} 8 & 1 & 6 \\ 3 & 5 & 7 \\ 4 & 9 & 2 \end{matrix}$$

Donc :

$$A * h = \begin{matrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{matrix} * \begin{matrix} 8 & 1 & 6 \\ 3 & 5 & 7 \\ 4 & 9 & 2 \end{matrix} = \begin{matrix} 7 & 14 & 20 & 9 \\ 8 & 20 & 16 & 13 \\ 17 & 22 & 29 & 17 \\ 12 & 15 & 14 & 4 \end{matrix}$$

2.4 Filtrage linéaire des images sous MATLAB

Filtrage des images, soit par corrélation ou convolution, peut être effectué à l'aide de la fonction de boîte à outils **imfilter**. Cet exemple filtre une image avec un filtre 5-par-5 contenant des poids égaux, un tel filtre est souvent appelé un filtre de moyenne :

Exemple 2.4.1 *I = imread('coins.png');*

h = ones(5,5) / 25;

I2 = imfilter(I,h);

subplot(1,2,1)

imshow(I), title('Image Originale');

subplot(1,2,2),

imshow(I2), title('Image Filtré')

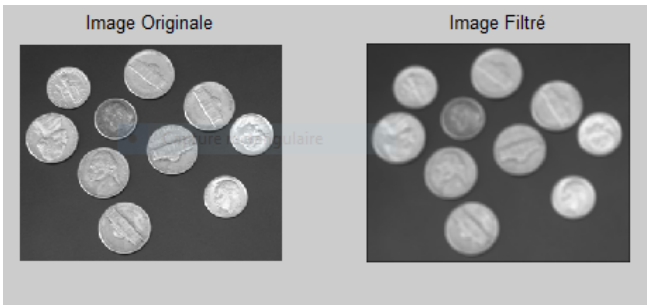

Figure14:Résultat de convolution

2.5 Traitement des bords

Quand les valeurs du noyau tombent à l'extérieur de l'image , une partie du noyau de convolution ou de corrélation est habituellement hors du bord de l'image, selon les cas suivants :

Premier cas :

Remplissage zéro des pixels extérieurs(voir[7])

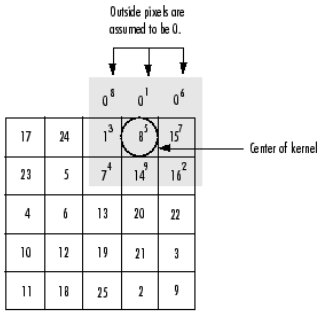

Figure15 : Remplissage du la matrice

La fonction **imfilter** remplit normalement ces pixels d'image hors ligne en supposant qu'ils sont 0, c'est ce qu'on appelle le rembourrage zéro est illustré comme suit :

Exemple 2.5.1 *I = imread('houda.jpg');*

i=rgb2gray(I);

h = ones(5,5) / 25;

I2 = imfilter(i,h);

imshow(i), title('Image Originale');

figure, imshow(I2), title('Image filtré avec Black and Border ')

Lorsque on filtre une image, le rembourrage zéro peut entraîner une bande sombre autour du bord de l'image, comme indiqué dans cet exemple :

Figure 16: Le filtre Black and Border

Deuxième cas

Pixels de limites répliqués

Pour éliminer les artefacts de rembourrage à zéro autour du bord de l'image, **imfilter** offre une méthode de rembourrage de limite alternative appelée réplication de bordure. Dans la réplication de bordure, la valeur de tout pixel en dehors de l'image est déterminée en répliquant la valeur à partir du pixel de bordure le plus proche. C'est illustré dans la figure suivante :

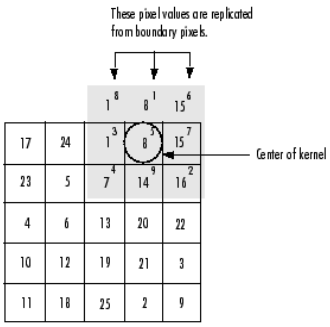

Figure 17: Remplissage de la matrice

Pour filtrer en utilisant la réplication de bordure, passons l'argument optionnel supplémentaire 'replicate' à **imfilter**

Exemple 2.5.2 *I3 = imfilter(i,h,'replicate');*

figure, imshow(I3);

title(' Image Filtré avec Border Replication')

Figure 18: Le filtre Border Replication

2.6 Filtre Gaussien sous MATLAB

Définition 2.6.1 *Le filtre gaussien est un filtre linéaire isotrope spécial avec des propriétés mathématiques bien précises. La fonction Gaussienne est très connu dans la nature. La fonction Gaussienne est aussi souvent utilisée dans les distributions statistiques, elle est définie par la fonction.(voir[5])*

$$G(x) = \frac{1}{\sigma\sqrt{2\Pi}} \exp -\frac{x^2}{2\sigma^2}$$

Sous matlab le filtrage ce fait comme suit :

Exemple 2.6.1 *i=imread('coins.png');*

h=fspecial('gaussian',[5 5], 2);

I2=imfilter(i,h,'conv');

h1=fspecial('gaussian',[5 5], 7);

I3=imfilter(i,h1,'conv');

```
h2=fspecial('gaussian',[5 5], 14);
I4=imfilter(i,h2,'conv');
subplot(2,2,1)
imshow(i)
title('Image Originale')
subplot(2,2,2)
imshow(I2)
title('σ = 2')
subplot(2,2,3)
imshow(I3)
title('σ = 7')
subplot(2,2,4)
imshow(I4)
title('σ = 14')
```

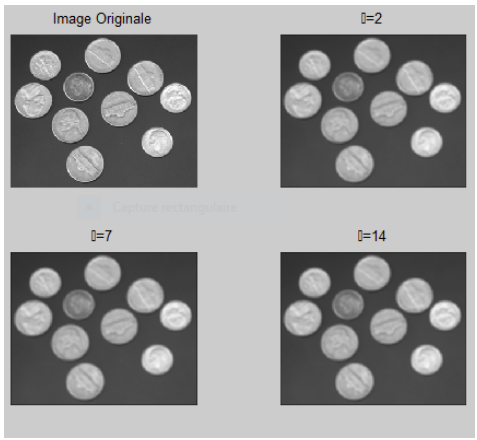

Figure 19: Le filtre de Gaussien

2.7 Filtrage d'une image avec des types de filtre prédéfinis

La fonction **fspecial** produit plusieurs types de filtres prédéfinis, sous forme de noyaux de corrélation, après avoir créé un filtre avec **fspecial**, nous pouvons l'appliquer directement à nos données d'image en utilisant **imfilter**. Cet exemple illustre l'application d'un filtre de masquage sans partage à une image en niveaux de gris.(voir[7])

Exemple 2.7.1 *I = imread('houda.jpg');*

rgb=rgb2gray(I);

h = fspecial('unsharp');

I2 = imfilter(rgb,h);

subplot(1,2,1)

imshow(rgb), title('Image Originale')

subplot(1,2,2)

imshow(I2), title('Image Filtré')

Figure 20: Le filtre Unsharp

2.8 Le bruit sous MATLAB

Nous pouvons définir le bruit pour toute dégradation du signal d'image, causée par une perturbation externe. Si une image est envoyée électroniquement d'un endroit à l'autre,

via une transmission par satellite ou sans fil, ou via un câble en réseau, nous pouvons nous attendre à ce que des erreurs se produisent dans le signal de l'image. Ces erreurs apparaîtront

sur la sortie de l'image de différentes façons en fonction du type de perturbation dans le signal.Habituellement, nous savons quel type d'erreurs attendre, et donc le type de bruit sur l'image par conséquent, nous pouvons choisir la méthode la plus appropriée pour réduire les effets. Le nettoyage d'une image corrompue par le bruit est donc une zone importante de restauration d'image.(voir[5])

Comprendre les sources du bruit dans les images numériques

Il existe plusieurs façons dont le bruit peut être introduit dans une image, selon la façon dont l'image est créé. Par exemple :

1. Si l'image est balayée sur une photo réalisée sur film, le grain de film est une source de bruit. Le bruit peut également être le résultat d'un dommage au film,Ou être introduit par le scanner lui-même.

2. Si l'image est acquise directement dans un format numérique, le mécanisme de collecte des données (tel qu'un détecteur CCD) peut introduire du bruit.

3. La transmission électronique des données d'image peut introduire du bruit.

Pour simuler les effets de certains des problèmes énumérés ci-dessus, la boîte à outils fournit la fonction **imnoise**, que nous pouvons utiliser pour ajouter différents types du bruit à une image.Les exemples de cette section utilisent cette fonction.

Les types du bruit

-Bruit Poivre et sel :

Définition 2.8.1 *Le bruit impulsionnel, également appelé sel et poivre est une dégradation de l'image. Ce dernier d'ordre n est obtenu en ajoutant n pixels blancs et n pixels noirs aléatoirement dans une image.*

Exemple 2.8.1 *I = imread('star.jpg');*

rb=rgb2gray(I);J = imnoise(rb,'salt & pepper',0.02);

imshow(J),title('Bruit sel et poivre')

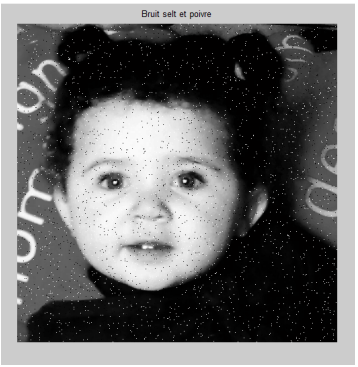

Figure 21: Le bruitage

-Bruit Gaussien :

Définition 2.8.2 *Loi de distribution Gaussienne de variance σ et moyenne μ : Le bruit Gaussien est obtenu en ajoutant à chaque pixel une valeur aléatoire suivant une loi de probabilité Gaussienne.*

$$G_{\sigma,\mu}(s) = \frac{1}{\sqrt{2\pi}\sigma} \exp^{-\frac{(s-\mu)^2}{2\sigma^2}}$$

Exemple 2.8.2 *I = imread('star.jpg');*

rb=rgb2gray(I); J = imnoise(rb,'gaussian',0,0.025);

imshow(J),title('Bruit Gaussien')

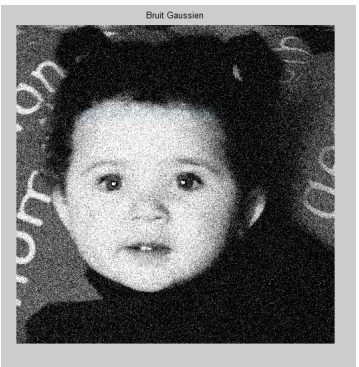

Figure 21:Le bruitage

2.9 Débruitage sous MATLAB

Suppression du bruit par filtrage linéaire

On peut utiliser un filtrage linéaire pour supprimer certains types de bruit.certains filtres, comme la moyenne ou les filtres gaussiens sont appropriés à cet effet. Par exemple, un filtre de calcul de moyenne est utile pour éliminer le bruit de grain d'une photo parce que chaque pixel est réglé sur la moyenne des pixels dans son quartier, les variations locales causées par les grains sont réduites.(voir[9])

- Lire dans l'image et l'afficher :

I = imread('houda.jpg');
rb=rgb2gray(I);
figure,
imshow(rb)

Figure22:L'image Originale

- Ajoutons du bruit.

J = imnoise(rb,'salt & pepper',0.02);
figure,
imshow(J)

Figure 23:Le bruit sel et poivre

- Filtrer l'image bruitée avec un filtrage moyen et regarder le resultat

K = filter2(fspecial('average',3),J)/255;
figure,

imshow(K)

Figure 24:Image filtré

Suppression du bruit par le filtrage médian

Le filtrage médian est similaire à l'utilisation d'un filtre de moyenne, en ce que chaque pixel de sortie est réglé sur une moyenne des valeurs de pixels dans le voisinage du pixel d'entrée correspondant. Cependant, avec un filtrage médian, la valeur d'un pixel de sortie est déterminée par la médiane des pixels de quartier, plutôt que le moyen. La médiane est beaucoup moins sensible que les valeurs moyennes à extrêmes (appelées valeurs aberrantes).

Le filtrage médian est donc mieux en mesure de supprimer ces valeurs aberrantes sans réduire la netteté de l'image. La fonction **medfilt2** implémente le filtrage médian.(voir[5])

- Lire dans l'image et l'afficher.

I = imread('houda.jpg');
rb=rgb2gray(I);
figure,
imshow(rb)

Figure 25:L'image Originale

- Ajoutons du bruit.

J = imnoise(rb,'salt & pepper',0.02);
figure,
imshow(J)

Figure 26 :L'image bruité

- Maintenant utiliser le fitre median

L = medfilt2(J,[3 3]);

figure,

imshow(L)

Figure 27:Image Filtré

Remarque 2.9.1 *Nous pouvons voir que le filtre médian est plus efficace pour l'image que le filtre de moyenne.*

Suppression du bruit par filtrage adaptatif

La fonction **wiener2** applique un filtre Wiener (un type de filtre linéaire) à une image de manière adaptative, s'ajuster à la variance de l'image locale. Où la variance est grande, **wiener2** effectue peu de lissage. Lorsque la variance est faible, **wiener2** effectue plus de lissage.

Cette approche produit souvent de meilleurs résultats que le filtrage linéaire. Le filtre adaptatif est plus sélectif qu'un filtre linéaire comparable, en conservant les bords et d'autres parties haute fréquence d'une image**. Wiener2** cependant, nécessite plus de temps de calcul que le filtrage linéaire.(voir[5])

Wiener2 fonctionne mieux lorsque le bruit est de puissance constante ("blanc"), comme le bruit gaussien. L'exemple ci-dessous s'applique **wiener2** à une image qui a eu un bruit gaussien ajouté.

- Lire dans une image. Parce que l'image est une image truecolor, l'exemple la convertit en niveaux de gris.

RGB = imread('houda.jpg');
I = rgb2gray(RGB);

- L'exemple ajoute du bruit gaussien à l'image puis affiche l'image. Parce que l'image est assez grande, la figure ne montre qu'une partie de l'image

J = imnoise(I,'gaussian',0,0.025);
imshow(J)

Figure 28:Image Bruité

- Retirons le bruit, en utilisant la fonction **wiener2**. Encore une fois, le chiffre ne montre qu'une partie de l'image

K = wiener2(J,[5 5]);
figure, imshow(K)

Figure 29:Le filtre Wiener

2.10 Filtrage d'une partie d'image sous MATLAB

Le filtrage d'une région d'intérêt (ROI) est le processus d'application d'un filtre à une région dans une image, où un masque binaire définit la région.Par exemple, nous pouvons appliquer un filtre de réglage d'intensité à certaines régions d'une image.(voir[7])

Pour filtrer un ROI dans une image, utilisons la fonction **roifilt2**. Lorsque nous appelons **roifilt2**, nous spécifions:

- Entrer l'image en niveaux de gris à filtrer.
- Image de masque binaire qui définit le ROI.
- Filtre (soit un filtre 2-D ou une fonction).

Cet exemple utilise un filtrage masqué pour augmenter le contraste d'une région spécifique d'une image:

1. Lire dans une image :

   I = imread('pout.tif');

2. Créer le masque :

   Cet exemple utilise le masque BW créé par la méthode createMask dans la section Création d'un masque binaire. La région d'intérêt spécifiée est le visage de l'enfant.

   img = imread('pout.tif');

   h_im = imshow(img);

   e = imellipse(gca,[55 10 120 120]);

   BW = createMask(e,h_im);

3. Utilisons **fspecial** pour créer le filtre:

   h = fspecial('unsharp');

4. Appelons **roifilt2**, spécifiant le filtre, l'image à filtrer et le masque:

I2 = roifilt2(h,I,BW);

imshow(I)

figure, imshow(I2)

Figure 30: Image Originale et Filtré

2.11 Restauration d'image flou ou dégradée sous MATLAB

Le flou ou la dégradation, d'une image peut être causée par de nombreux facteurs:

- Mouvement au cours du processus de capture d'image, par l'appareil photo ou, lorsque de longues durées d'exposition sont utilisés, par le sujet.
- Distorsion lumineuse dispersée en microscopie confocale.

Modèle de restauration

une image floue ou dégradée peut être décrite approximativement par cette équation

$$g = Hf + n$$

Où

g L'image floue.

H l'opérateur de distorsion, également appelé fonction de propagation de points(point spread function) (PSF).

f L'image véritable originale.

n Le bruit additif, introduit lors de l'acquisition d'image.(voir[9])

Importance du PSF

Sur la base de ce modèle, la tâche fondamentale de restauration est de déconvolution de l'image floue avec le PSF qui décrit exactement la distorsion.

La déconvolution est le processus d'inversion de l'effet de la convolution.(voir[9])

Pour illustrer, cet exemple prend une image claire et le délire délibérément en le convainquant avec un PSF. L'exemple utilise la fonction **fspecial** pour créer un PSF qui simule un flou de mouvement, en spécifiant la longueur du flou en pixels (LEN = 31) et l'angle du flou en degrés (THETA = 11). Une fois que le PSF est créé, l'exemple utilise la fonction **Imfilter** pour convolution du PSF avec l'image d'origine I, pour créer l'image floue.

Exemple 2.11.1 *I = imread('peppers.png');*

I = I(60+[1:256],222+[1:256],:); % Recadrer l'image

figure; imshow(I); title(' Image Originale');

Figure 31: Image Originale

LEN = 31;

THETA = 11;

PSF = fspecial('motion',LEN,THETA); % créer PSF

Blurred = imfilter(I,PSF,'circular','conv');

figure; imshow(Blurred); title(' Image Flou');

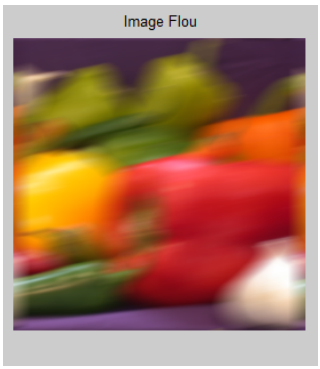

Figure 32: Image Flou

Restauration avec le filtre Wiener

Utilisons la fonction **deconvwnr** pour restaurer une image à l'aide du filtre Wiener. La déconvolution de Wiener peut être utilisée efficacement lorsque les caractéristiques de fréquence de l'image et du bruit additif sont connues, au moins dans certains cas. En l'absence de bruit, le filtre Wiener se réduit au filtre inversé idéal.

Cet exemple restaure l'image floue crée dans le Modèle de suppression, en spécifiant la même fonction PSF qui a été utilisée pour créer le flou. Cet exemple illustre l'importance de connaître le PSF, la fonction qui a causé le flou. Lorsque nous connaissons le PSF exact, les résultats de la désaffectation peuvent être très efficaces(voir[2])

- Restorons l'image

Exemple 2.11.2 *wnr1 = deconvwnr(Blurred,PSF);*

figure;

imshow(wnr1);

Figure 33: Image Restauré

Restauration avec un filtre régulier

Utilisons la fonction **deconvreg** pour restaurer une image à l'aide d'un filtre normalisé. Un filtre régulé peut être utilisé efficacement lorsque des informations limitées sont connues sur le bruit additif.

Pour illustrer, cet exemple simule une image floue en convertissant un filtre PSG gaussien avec une image (utilisant imfilter). Le bruit additif dans l'image est simulé en ajoutant du bruit gaussien de variance V à l'image floue (en utilisant imnoise) .(voir[2])

- Lisons une image dans l'espace de travail MATLAB. L'exemple utilise le recadrage pour réduire la taille de l'image à restaurer. Ce n'est pas une étape requise dans les opérations de restauration.

Exemple 2.11.3 ***I = imread('tissue.png');***
I = I(125+[1:256],1:256,:);
figure; imshow(I); title(' Image Originale');

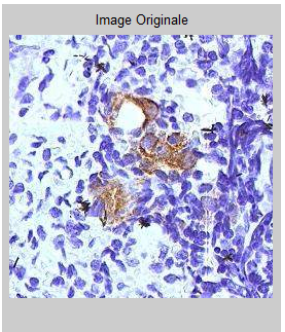

Figure 34:Image Originale

- créer le PSF

PSF = fspecial('gaussian',11,5);

- Créons un flou simulé dans l'image et ajoutons du bruit

Blurred = imfilter(I,PSF,'conv');

V = .02;

BlurredNoisy = imnoise(Blurred,'gaussian',0,V);
figure;imshow(BlurredNoisy);title('Image Flou et Bruité');

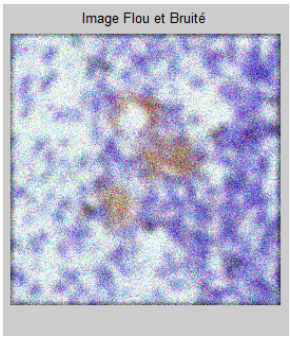

Figure 35:Image Flou et Bruité

- Utilisons **deconvreg** pour restaurer l'image, en spécifiant le PSF utilisé pour créer le flou et la puissance de bruit, NP.

NP = V*prod(size(I));
[reg1 LAGRA] = deconvreg(BlurredNoisy,PSF,NP);
figure,imshow(reg1),title('Image Restauré');

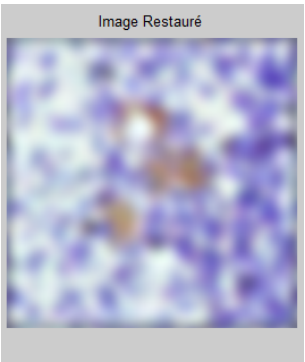

Figure 36: Image Restauré

Utilisation de la fonction deconvlucy pour restaurer une image

Pour illustrer une simple utilisation de **deconvlucy**, cet exemple simule une image brouillée et bruyante en convertissant un PSF de filtre gaussien avec une image (en utilisant **imfilter**) puis en ajoutant du bruit Gaussien de variance V à l'image floue (en utilisant **imnoise**):

- Lisons une image dans l'espace de travail MATLAB. (L'exemple utilise le recadrage pour réduire la taille de l'image à restaurer. Ceci n'est pas une étape requise dans les opérations de restauration.) (voir[2])

Exemple 2.11.4 ***I = imread('board.tif');***
I = I(50+[1:256],2+[1:256],:);
figure;imshow(I);title('Image Originale');

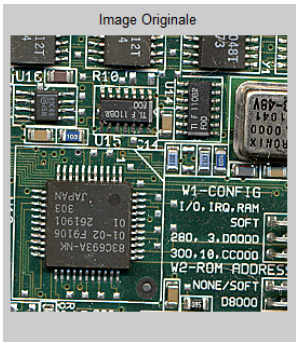

Figure 37: Image Originale

- créer le PSF

PSF = fspecial('gaussian',5,5);

- Créons un flou simulé dans l'image et ajoutons du bruit.

Blurred = imfilter(I,PSF,'symmetric','conv');

V = .002;
BlurredNoisy = imnoise(Blurred,'gaussian',0,V);
figure;imshow(BlurredNoisy);title('Image Flou et Bruité');

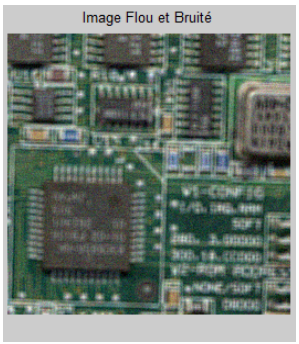

Figure 38:Image Flou et Bruité

- Utilisons **deconvlucy** pour restaurer l'image floue et bruyante, en spécifiant le PSF utilisé pour créer le flou et en limitant le nombre d'itérations à 5 (la valeur par défaut est 10).

luc1 = deconvlucy(BlurredNoisy,PSF,5);

figure; imshow(luc1);
title('Image Restauré');

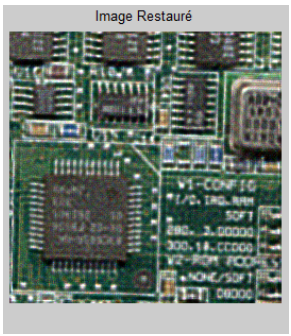

Figure 39: Image Restauré

Chapitre 3

La détection des contours dans les images

3.1 Principe du détection des contours

Introduction:

La détection de contours dans les images a débuté dans les années 80, des approches plus systématiques ont été mises en place par Marr [Marr et Hildreth, 1980], puis Canny [Canny, 1986], pour obtenir des contours plus significatifs. Ces travaux ont abouti à une bonne compréhension de ce qu'il faut faire pour détecter les contours, mais la définition mème des contours demeure trés vague, ce qui rend ces techniques encore peu efficaces sur un problème concret. De plus il faut noter que le problème de détection de contours est généralement mal posé(au sens de la résolution des systèmes) [Torre et Poggio, 1986]. Les seuls modèles de contours utilisables sont ceux de contours idéalisés, comme ceux présentés sur la figure 40, ils sont bien loin de la réalité. C'est pourquoi, mème si de trés gros progrés ont été accomplis dans ce domaine, les techniques d'estimation du gradient proposées dans les années 70-80 restent souvent encore employées en concurrence de techniques plus modernes. Une excellente référence à ce problème est l'ouvrage collectif [Cocquerez et Philipp, 1996]. Plusieurs articles existent sur la comparaison des performances de détecteurs de contours, en particulier [Palmer et al., 1996, Heath et al., 1997].

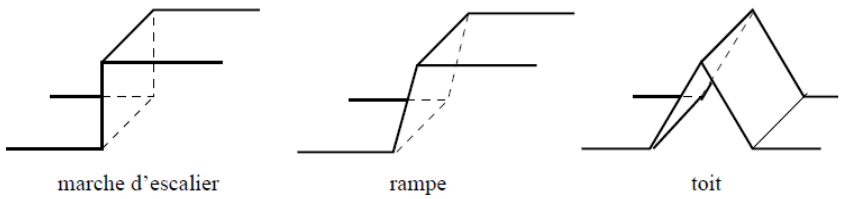

Figure 40: Quelques modèles de contours Le plus utilisé est celui en marche d'escalier.

Nous présenterons dans la suite l'utilisation de MATLAB dans la détection des contours, celle ci est déterminée par une technique mathématique, qui consiste à chercher le maximum du module du gradient ou bien les zéros du Laplacien. Le calcul des valeurs du contour via Matlab se base sur des algorithmes de détection de contours permettent d'identifier les limites des objets d'une image. Ces algorithmes comprennent les méthodes Sobel, Prewitt, Roberts, Canny et laplacien de Gauss. La méthode Canny peut détecter les véritables contours faibles sans être induit en erreur par le bruit.

La détection du bord est une technique de traitement d'image pour trouver les limites d'objets dans les images. Il fonctionne en détectant les discontinuités dans la luminosité. La détection du bord est utilisée pour la segmentation d'image et l'extraction de données dans des domaines tels que le traitement d'image, la vision par ordinateur et la vision mécanique.

Définition du contours

Définition 3.1.1 *Le contour est définie en tant que variation (ou discontinuité) de l'intensité d'image ,Si* $f(x, y)$ *une image, le contours est le lieu des fortes variations de* f *, et qui peut etre aussi décrit par le zéros Laplacien.Le principe classique de la détection de contours repose sur l'étude des dérivées de la fonction d'intensité dans l'image : les extréma locaux du gradient de la fonction d'intensité et les passages par zéro du laplacien on peut illustrer cela par les graphes suivants:*

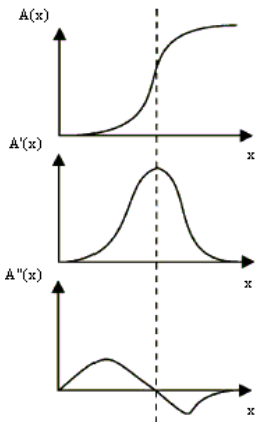

Figure 41: les graphes du gradient

Le but de détection du contour :

Le but de la détection de contours est de repérer les points d'une image numérique qui correspondent à un changement brutal de l'intensité lumineuse. La détection des contours d'une image réduit de manière significative la quantité de données et élimine les informations qu'on peut juger moins pertinentes, tout en préservant les propriétés structurelles importantes de l'image. Il existe un grand nombre de méthodes de détection de l'image. Dans ce qui suit quelques méthodes seront exposées .

3.2 Les méthodes de détection du contour sous Matlab

3.2.1 Méthode de Sobel:

Le principe de ce filtre est que l'opérateur calcule le gradient de l'intensité de chaque pixel. Ceci indique la direction de la plus forte variation du clair au sombre, ainsi que le taux de échangement dans cette direction. On connaît alors les points de changement soudain de luminosité, correspondant probablement à des bords, ainsi que l'orientation de ces bords.

L'opérateur utilise des matrices de convolution, la matrice (généralement de taille 3×3).

Soit A l'image source, G_X et G_Y deux images qui en chaque point contiennent des approximations respectivement de la dérivée horizontale et verticale de chaque point. Ces images sont calculées comme suit:

$$G_X = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * A \qquad ET \qquad G_Y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * A$$

En chaque point, les approximations des gradients horizontaux et verticaux peuvent être combinées comme suit pour obtenir une approximation de la norme du gradient:

$$|G| = \sqrt{G_X^2 + Gy^2}$$

Méthode de Sobel sous Matlab:

- BW = edge(I,'sobel'), spécifie la méthode Sobel.
- BW = edge(I,'sobel',thresh), spécifie le seuil de sensibilité pour la méthode Sobel .
- BW = edge(I,'sobel',thresh,direction), spécifie la direction de détection pour la méthode Sobel. Direction est une chaîne spécifiant s'il faut rechercher les bords «horizontal» ou «vertical» ou «les deux» (la valeur par défaut).
- [BW, thresh] = edge (I, 'sobel', ...) renvoie la valeur de seuil.
- [BW, thresh, gv, gh] = edge (I, 'sobel', ...) renvoie les réponses de bord verticales et horizontales aux opérateurs de dégradés de Sobel. Cet exemple illustre la detection du contour par la méthode de Sobel.

Exemple 3.2.1 *On implémente alors le script suivant :*

```
I=imread('houda.jpg');
J=rgb2gray (I);
J=double(J)/255.0;
seuil=0.8;
```

```
H=fspecial('sobel');
V=-H';
Gh=filter2(H,J);
Gv=filter2(V,J);
G=sqrt(Gh.*Gh + Gv.*Gv);
Gs1=(G>seuil*4/3);
imshow(J);
subplot(2,2,1);imshow(Gh);
subplot(2,2,2);imshow(Gv);
subplot(2,2,3);imshow(G);
subplot(2,2,4);imshow(Gs1);
```

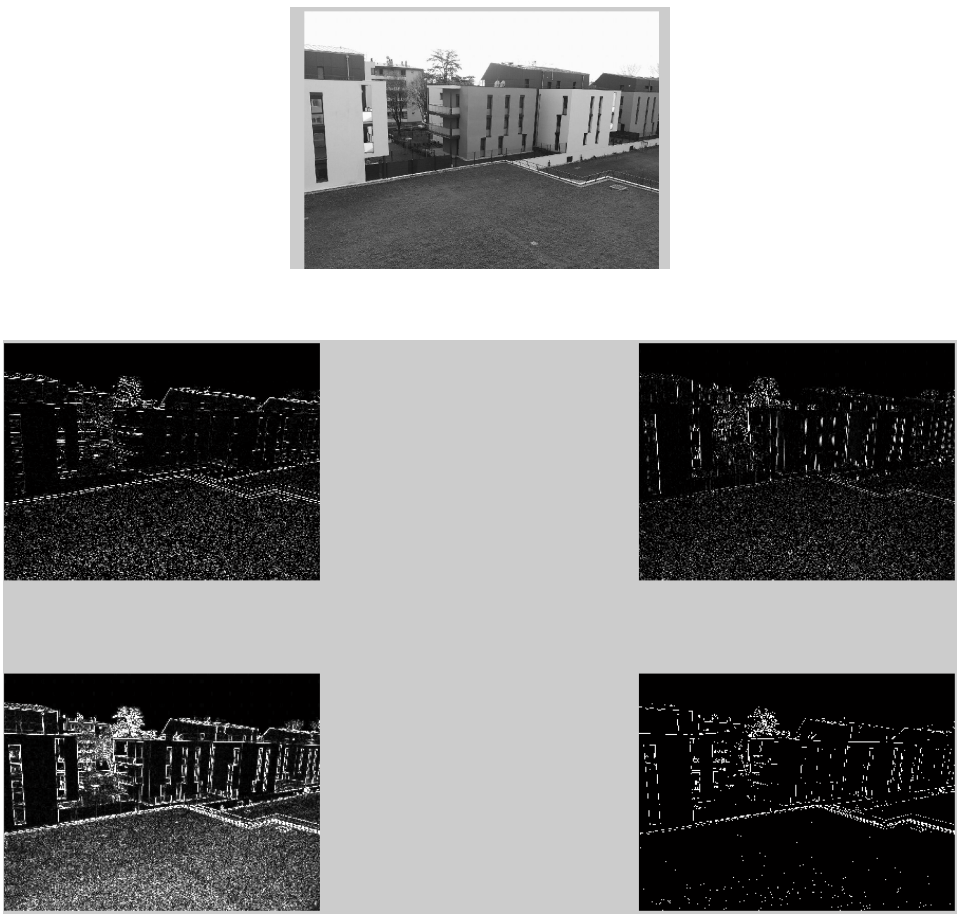

Figure 42: Les résultats du filtre sobel

3.3 Méthode de Prewitt:

La matrice qui correspond au filtrage horizontal, faisant ressortir essentiellement les contours verticaux, selon l'opérateur de Prewitt, s'écrit $H_X = [-1\ 0\ 1]$ tandis que la matrice verticale H_Y est sa transposée. Les deux convolutions avec le tableau de valeurs initiales créent deux tableaux $G_X$et G_Y à l'origine du tableau G sur lequel on peut localiser les maximums. Afin de réaliser ce filtre sous Matlab on applique le code suivant :

Méthode de Perwitt sous matlab:

- BW = edge (I, 'prewitt') spécifie la méthode Prewitt.
- BW = edge(I,'prewitt',thresh) spécifie le seuil de sensibilité pour la méthode Prewitt.Bord ignore tous les arêtes qui ne sont pas plus fortes que le seuil. Si vous ne spécifiez pas le seuil, ou si le seuil est vide ([ ]), le bord choisit la valeur automatiquement.
- BW = edge(I,'prewitt',thresh,direction) spécifie la direction de détection pour la méthode Prewitt.Direction est une chaîne spécifiant s'il faut rechercher les bords «horizontal» ou «vertical» ou «les deux» (la valeur par défaut).
- [BW, thresh] = edge (I, 'prewitt', ...) renvoie la valeur de seuil.([7])

Exemple 3.3.1 *I=imread('houda.jpg');*

J=rgb2gray (I);

J=double(J)/255.0;

seuil=0.8

H=fspecial('prewitt');

V=-H';

Gh=filter2(H,J);

Gv=filter2(V,J);

*G=sqrt(Gh.*Gh + Gv.*Gv);*

Gs=(G>seuil);

imshow(J);

subplot(2,2,1)imshow(Gh);

subplot(2,2,2)imshow(Gv);

subplot(2,2,3)imshow(G);

subplot(2,2,4)imshow(Gs);

On visualise le résultat sur la figure:

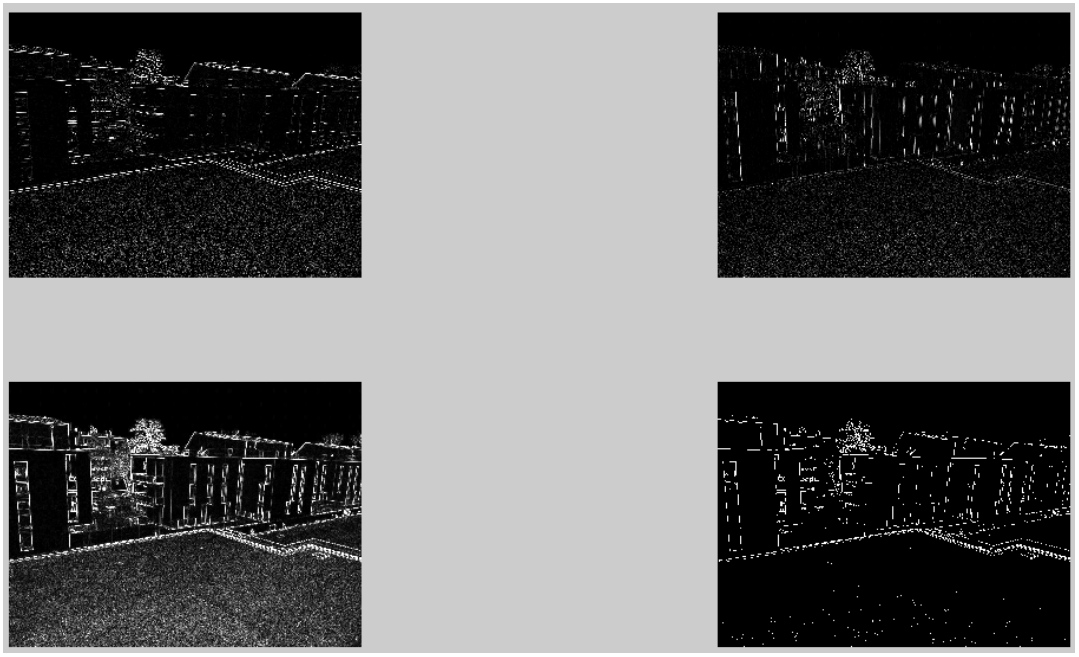

Figure 43: Les résultats pour le filte perwitt

3.4 Méthode de Roberts:

Les filtres de Roberts sont une approche discrète de la dérivée de pas 1 d'une fonction: le gradient de cette fonction. Si $I(x, y)$ représente un pixel dans une image, alors les amplitudes des gradients en X et en Y peuvent s'écrire respectivement:

$$G_X = I(X + 1, y) - I(X, Y)$$

$$G_Y = I(X, y + 1) - I(X, Y)$$

Cela revient a convoluer l'image avec les deux filtres $R_X = [-1\ 1]$ etR_Y =transpose($[-1$ 1]) . L'amplitude du gradient peut être alors calculée de plusieurs façons:

$$G_1(X, Y) = \text{sqrt}(G_X^2 + G_Y^2)$$

$$G_2(X, Y) = \max(\text{abs}(G_X) + \text{abs}(G_y))$$

ou

$$G_3(X, Y) = \text{abs}(G_X) + \text{abs}(G_y)$$

Et la direction du gradient est donneée par:

$$D(X, Y) = \arctan(G_Y/G_X)$$

Ou le bruit peut aussi être une brusque variation locale des niveaux de gris : ces filtres sont donc très sensibles au bruit car ils accentuent, par dérivation, le bruit présent dans l'image. De plus, ces filtres donneront un contour épais si celui-ci est un contour de type "rampe". Le code utilisé est le suivant :

Méthode de Roberts sous matlab:

- BW = edge (I, 'roberts') spécifie la méthode Roberts.

- BW = edge(I,'roberts',thresh) spécifie le seuil de sensibilité pour la méthode Roberts.Bord ignore tous les arêtes qui ne sont pas plus fortes que le seuil. Si vous ne spécifiez pas le seuil, ou si le seuil est vide ([ ]), le bord choisit la valeur automatiquement.

- BW = Edge (I, 'roberts', ..., options) où les options peuvent être la chaîne de texte 'Éclaircie' ou 'Ne pas compter'.Lorsque vous spécifiez 'amincissement', ou ne spécifiez pas de valeur, l'algorithme applique l'amincissement du bord. La spécification de l'option 'nothinning' peut accélérer le fonctionnement de l'algorithme en ignorant l'étape supplémentaire d'éclaircie du bord.

- [BW,thresh] = edge(I,'roberts',...) .renvoie la valeur de seuil.

- [BW,thresh,g45,g135] = edge(I,'roberts',...) Renvoie des réponses de bordure de 45 degrés et de 135 degrés aux opérateurs de gradient Roberts.([7])

Cet exemple illustre la detection du contour par la méthode de Roberts. ;

Exemple 3.4.1 *I=imread('houda.jpg');*

J=rgb2gray (I);

J=double(J)/255.0;

seuil=0.8;

a=[1 0;0 -1];

b=[0 1;-1 0]Ga=filter2(a,J);

Gb=filter2(b,J);

*G=sqrt(Ga.*Ga + Gb.*Gb);*

*Gs1=(G>seuil*1/3);*

imshow(J);

subplot(2,2,1);imshow(Ga);

subplot(2,2,2);imshow(Gb);

subplot(2,2,3);imshow(G);

subplot(2,2,4);imshow(Gs1);

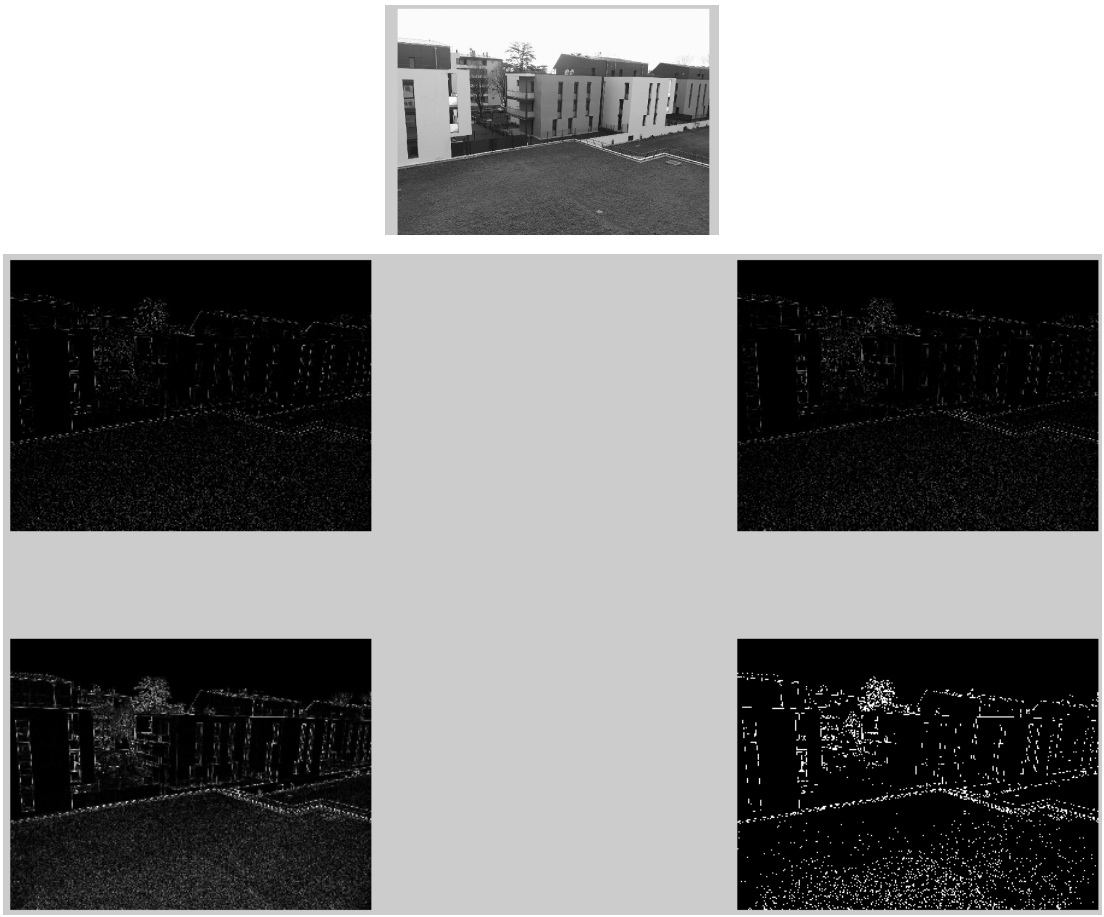

Figure 44: Les résultats du filtre Roberts

3.5 Méthode de Canny:

Le filtre de Canny (ou détecteur de Canny) est utilisé en traitement d'images pour la détection des contours. L'algorithme a été conçu par John Canny en 1986 pour être optimal suivant trois critères clairement explicités :

1. bonne détection : faible taux d'erreur dans la signalisation des contours,
2. bonne localisation : minimisation des distances entre les contours détectés et les contours réels,
3. clarté de la réponse : une seule réponse par contour et pas de faux positifs

Méthode de canny sous matlab:

- BW = edge (I, 'canny') spécifie la méthode Canny.
- BW = edge (I, 'canny', thresh) spécifie les seuils de sensibilité pour la méthode de Canny.Le seuil est un vecteur à deux éléments dans lequel le premier élément est le seuil bas, et le second élément est le seuil élevé. Si vous spécifiez un scalaire pour le seuil, cette valeur scalaire est utilisée pour le seuil élevé et 0,4 * seuil est utilisé pour le seuil bas. Si vous ne spécifiez pas le seuil, ou si le seuil est vide ([]), le bord choisit les valeurs faibles et élevées automatiquement.La valeur du seuil est relative à la valeur la plus élevée de la grandeur de gradient de l'image.
- BW = edge (I, 'canny', thresh, sigma) spécifie la méthode Canny, en utilisant sigma comme écart type du filtre gaussien.Sigma est 1; La taille du filtre est choisie automatiquement, selon le sigma.
- [BW, thresh] = edge (I, 'canny', ...) renvoie les valeurs de seuil en tant que vecteur à deux éléments.([9])

Exemple 3.5.1 *J = imread('E:\houda.jpg');*
I=rgb2gray (J);
BW2 = edge(I,'canny');

subplot(1,2,1);imshow(I)
subplot(1,2,2),imshow(BW1)

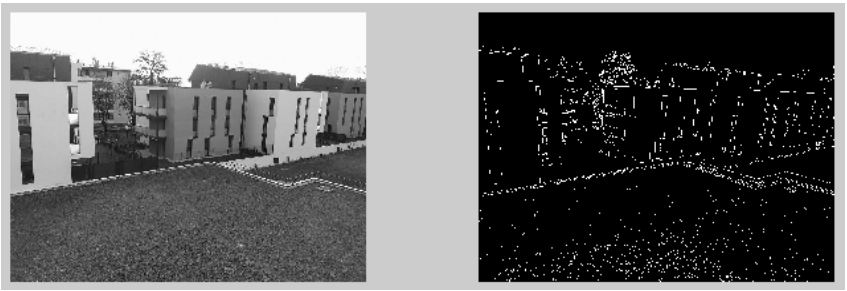

Figure 45: Les résultats du filtre canny

Résumé des méthodes utilisées:

BW = edge (I) prend une image en niveaux de gris ou binaire I en tant qu'entrée et renvoie une image binaire BW de la même taille que I, avec 1 où la fonction trouve des bords dans I et 0 ailleurs.

Par défaut, Edge utilise la méthode Sobel pour détecter les bords, mais ce qui suit fournit une liste complète de toutes les méthodes de recherche de bord supportées par cette fonction:

1. La méthode Sobel trouve des bords en utilisant l'approximation Sobel de la dérivée. Il renvoie des bords à ceux où le gradient de I est maximal.

2. La méthode Prewitt trouve des bords en utilisant l'approximation de Prewitt sur la dérivée. Il renvoie des bords à ceux où le gradient de I est maximal.

3. La méthode Roberts trouve des bords en utilisant l'approximation Roberts de la dérivée. Il renvoie des bords à ceux où le gradient de I est maximal.

4. La méthode Canny trouve des bords en recherchant les maxima locaux du gradient de I Le gradient est calculé en utilisant la dérivée d'un filtre gaussien. La méthode utilise deux seuils, pour détecter des bords forts et faibles, et inclut les bords faibles dans la sortie seulement s'ils sont connectés à des bords forts.Cette méthode est donc moins

susceptible que les autres d'être trompée par le bruit et plus susceptible de détecter de vrais arêtes faibles.

Les paramètres que vous pouvez fournir diffèrent selon la méthode que vous spécifiez. Si vous ne spécifiez pas de méthode, Edge utilise la méthode Sobel.

Exemple 3.5.2 *Cet exemple représente les différents méthodes de détection du contour qui sont prédifinis en Matlab.*

```
I =imread('coins.png');
A = edge(I,'sobel');
B =edge(I,'prewitt')
C =edge(I,'roberts')
D =edge(I,'canny')
imshow(I);figure
subplot(2,2,1);imshow(A)
subplot(2,2,2);imshow(B)
subplot(2,2,3);imshow(C)
subplot(2,2,4);imshow(D)
```

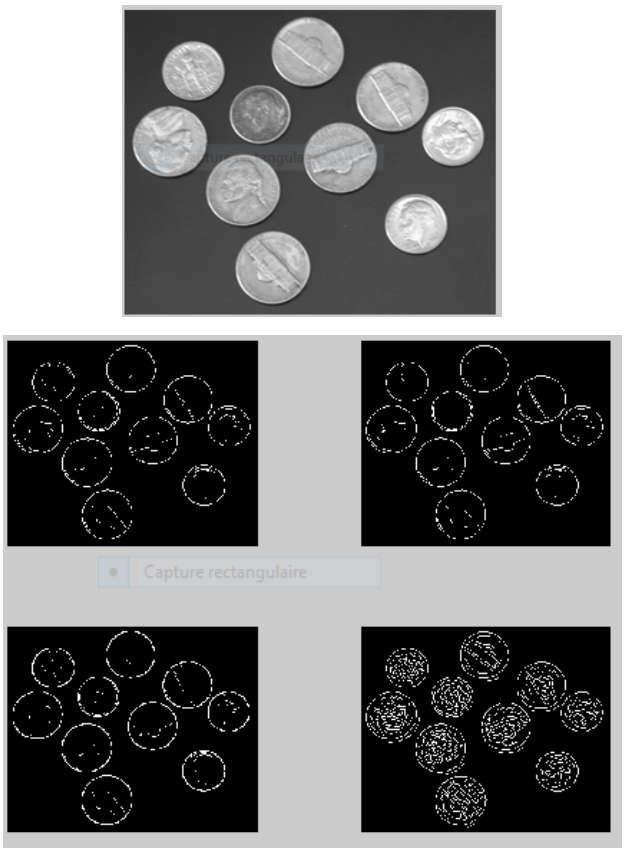

Figure 46: les différentes methodes de détections du contours

3.6 Traitement morphologique:

le traitement morphologique traite des outils d'extraction de composants d'image qui sont utiles dans la représentation et la description de la forme. L'analyse par morphologie mathématique vise à modifier la structure et la forme des objets de l'image, par exemple, pour séparer les objets, les discriminer en fonction de leur taille,remplir les trous, ...

Rappels

La transformation morphologique modifie la valeur d'un pixel de l'image en fonction de la valeur de ses voisins. Pour cela, on utilise un élément structurant, qui est un masque binaire. Il permet de prendre en compte le voisinage du pixel. Les traitements morphologiques sont définis à partir de deux opérations de base qui sont l'érosion et la dilatation.

3.6.1 Réduction de bruit par un filtre morphologique

En morphologie mathématique, filtrer c'est conserver l'image en supprimant certaines structures géométriques (en général implicitement définies par un ou plusieurs éléments structurants). Le filtre morphologique simplifie l'image en lui préservant la structure, mais il perd en général de l'information. Le filtre morphologique est stable et possède une classe d'invariance connue.

Le code correspondant à ce filtre est le suivant :

Exemple 3.6.1 *h=imread ('houda.jpg');*

s= strel('square',15) ;

i=imerode(h,s);

j=imdilate(h,s); figure(1);

subplot(1,3,1); imshow(h);

subplot(1,3,2); imshow(i);

subplot(1,3,3); imshow(j);

Figure 47: Un fitre morphologique

Conclusion

Dans ce travail, on s'est intéressé à traiter les image sous matlab. Nous avons introduit dans ce mémoire les notions de base qui servent de fondement à la compréhension de différentes techniques de traitement d'images. Plusieurs méthodes classiques de traitement ont été proposés dans la littérature, nous avons présenté quelques unes qui nous semble les plus courantes dans le processus du traitement et analyse d'image. Nous nous somme concentré sur les principales fonction du Matlab en traitement de l'image dans la boite à outils toolbox.

Aprés avoir donner quelques définitions et proriètés générales sur l'image. Nous nous sommes intéréssé, à traiter deux axes importants à savoir :

La restauration d'image sous MATLAB et la détection du contour .

Dans le premier axe, l'étude a été consacré à la restauration d'image et on particulier le filtrage linéaire et non linéaire, bruitage et débruitage et les différents types de la déconvolution .

Enfin dans le deuxième axe, une étude détaillée a été présentée concernant les différentes méthodes de la détection du contour .

Merci

Bibliographie

[1] B. Pesquet-Popescu,D. Matignon,F. Schmitt,H. Maitre,I. Bloch,M. Sigelle,Y. Gousseau, *Le traitement des images/tome 2/. Tupin/version 5.0/8 decembre 2005*

[2] *Matlab User's Guide .The Math works Inc,1995.*

[3] Gilles Burel, *Introduction au traitement d'images - simulation sous matlab19 octobre 2001de.*

[4] Jean-Thierry, *Lapresté, Introduction à MATLAB, Ellipses, 2005 (ISBN 2729824014).*

[5] M.Mokhtari et A.Mesbah *Apprendre et maitriser Matlab,Springer Verlag,1997.*

[6] *Sym function [archive] Documentation for the MATLAB Symbolic Toolbox*

[7] *http://www.mathworks.com/products/simulink-coder [archive] Simulink Coder*

[8] Richard Goering, *"Matlab edges closer to electronic design automation world [archive]," EE Times, 10/04/2004*

[9] *«http://www.mathworks.com/access/helpdesk/help/techdoc/matlab_ oop/brqzfut-1.html »*

الملخص:

الهدف من هذه الدراسة هو توضيح بالتفصيل الأوامر والتعليمات في مربع الأدوات المخصصة لمعالجة الصور، بما في ذلك معالجة الصور، وتصفية الخطية وغير الخطية استعادة الصورة وأخيرا الكشف عن ملامح. فأصبحنا مهتمين معالجة الصورة في ماتلاب.

ABSTRACT:

The objective of this study is to illustrate in detail the commands and instructions contained in the toolbox dedicated to image processing, including image manipulation, linear and non-linear filtering, image restoration and finally The detection of contours. So we were interested in processing image under matlab.

THE KEY WORKS: Image, Matlab, filter, noise, denoising, restore, contour detection ,,,,.

RÉSUMÉ :

L'objectif de cette étude est d'illustrer en detail les commandes et instructions contenues dans la boite à outils dédiée au traitement de l'image, notamment les manipulation des images, le filtrage linéaire et non linéaire, la restauration d'image et enfin la detection des contours. Donc on s'est intéressé à traiter les image sous matlab.

LES MOTS CLÉS : Image, Matlab , filtrer, bruiter,débruiter , restaurer, détection du contour ,,,,.