



Département d’Informatique

Licence Sciences et Techniques en Informatique

Option : Génie Logiciel

### Mémoire de Projet de Fin d’Etudes

Intitulé :

**Développement d’algorithmes d’extraction des contours**

**par Sobel et Canny**

Préparé par :

- AIT TOUDA Mohamed

- OUALILI Lamia

- OURAHOU Mohamed

Soutenu le 08 Juillet 2021 devant le Jury :

- |                      |                  |             |
|----------------------|------------------|-------------|
| - Pr. BOUDA Brahim   | FST d’Errachidia | Encadrant   |
| - Pr. AKSASSE Brahim | FST d’Errachidia | Examinateur |
| - Pr. BAATAOUI Aziz  | FST d’Errachidia | Examinateur |

# Dédicaces

A nos très chers parents :

Qui nous ont soutenu par leur amour et leurs efforts.

Qui nous ont toujours encouragé.

Pendant toute la période de nos études et qui n'ont épargné aucun effort pour répondre à nos exigences.

A tous nos amis.

A tous nos collègues à la FSTE.

A nos professeurs.

Pour tout le soutien que vous nous avez apporté.

nous vous disons MERCI.

A tous ceux que nous aimons et qui nous aiment, nous dédions ce travail...

## **Remerciements**

Nous remercions Dieu de nous avoir donné la volonté et le courage afin d'arriver à ce modeste travail.

Nous tenons à exprimer vivement notre profonde gratitude à notre encadrant Monsieur : **Bouda Brahim** pour sa confiance, ses encouragements, ses merveilles corrections et pour les conseils qu'il a apporté pour l'achèvement de ce projet.

Nous tenons également à remercier l'ensemble de membres de jury qui nous ont fait l'honneur de juger notre travail.

Nous présentons aussi nos remerciements à tous nos professeurs de la spécialité Génie logiciel.

Nous remercions avec beaucoup d'amour et respect nos chers parents, qui nous ont fournis au quotidien un soutien et autant de sacrifices pour qu'on puisse avancer dans nos études.

Dans l'impossibilité de citer tous les noms et pour finir nous tenons aussi à exprimer nos remerciements à tous ceux qui nous ont aidés de près ou de loin durant l'élaboration de notre mémoire de fin d'étude.

# Table des matières

<b>1</b>	<b>Introduction générale</b>	<b>7</b>
<b>2</b>	<b>Notions de Base</b>	<b>9</b>
2.1	Introduction . . . . .	9
2.2	Généralités sur le traitement d'images : . . . . .	9
2.2.1	Définition : . . . . .	9
2.2.1.1	Une image vectorielle : . . . . .	10
2.2.1.2	Image matricielle : . . . . .	11
2.2.1.3	Image à niveau de gris : . . . . .	11
2.2.1.4	Image Binaire : . . . . .	12
2.2.1.5	Image couleur : . . . . .	12
2.2.2	Echantillonnage : . . . . .	13
2.2.3	Quantification : . . . . .	13
2.2.4	Caractéristiques d'image : . . . . .	13
2.2.4.1	Pixels : . . . . .	13
2.2.4.2	Histogramme : . . . . .	14
2.2.4.3	Région : . . . . .	15
2.2.4.4	Bruit : . . . . .	15
2.2.4.5	Types de bruit : . . . . .	15
2.2.4.6	Contour : . . . . .	18
2.2.4.7	Convolution : . . . . .	18
2.2.5	Filtrage : . . . . .	18
2.2.5.1	Filtre linéaire : . . . . .	19
2.2.5.2	Filtres non-linéaires : . . . . .	19
2.2.6	Conclusion . . . . .	20
<b>3</b>	<b>Outils de développement</b>	<b>21</b>
3.1	<i>Qui est-ce que Matlab</i> : . . . . .	21
3.2	<i>Environnement Matlab</i> : . . . . .	21
3.3	Caractéristiques de Matlab : . . . . .	23
3.4	Les fonctions de MATLAB utilisées pour un traitement d'image : . . . . .	23

3.5	Détection de contours sous MATLAB :	25
3.6	Conclusion :	25
<b>4</b>	<b>Les méthodes d'extraction des contours</b>	<b>26</b>
4.1	Introduction . . . . .	26
4.2	Méthodes de détection de contours . . . . .	26
4.2.1	Méthodes dérivatives . . . . .	27
4.2.1.1	Opérateur de gradient . . . . .	27
4.2.2	Méthodes déformables . . . . .	28
4.2.2.1	Filtre de Sobel . . . . .	28
4.2.2.2	Implémentation avec MATLAB . . . . .	33
4.2.2.3	Comparaison des résultats . . . . .	33
4.2.2.4	filtre de prewitt . . . . .	34
4.2.2.5	filtre de Roberts . . . . .	35
4.2.3	Méthodes analytique . . . . .	36
4.2.3.1	Filtre de Canny . . . . .	36
4.2.3.2	Etapes d'opérateur de Canny : . . . . .	36
4.2.3.3	Exemple d'application : . . . . .	43
4.2.3.4	Les avantages et les inconvénients de filtre de Canny : . . . . .	45
4.2.3.5	Détecteur de Deriche . . . . .	45
4.2.3.6	Conclusion : . . . . .	45
<b>5</b>	<b>Application</b>	<b>47</b>
5.1	Introduction : . . . . .	47
5.2	Les fonctionnalités de l'application : . . . . .	47
5.3	Réalisation de l'interface : . . . . .	48
5.4	Description de l'interface : . . . . .	48
5.5	Conclusion . . . . .	53
<b>6</b>	<b>Conclusion générale</b>	<b>54</b>

# Table des figures

2.1	Image vectorielle . . . . .	10
2.2	Image matricielle . . . . .	11
2.3	Image à niveau de gris . . . . .	11
2.4	Image binaire . . . . .	12
2.5	Image couleur . . . . .	12
2.6	Échantillonnage . . . . .	13
2.7	Quantification . . . . .	13
2.8	Image numérique . . . . .	14
2.9	Exemple d'histogramme . . . . .	15
2.10	(a)image originale (b)image bruitée par un bruit gaussien . . . . .	16
2.11	(a)image originale (b)image bruitée par un bruit speckle . . . . .	17
2.12	(a)image originale (b)image bruitée par un bruit sel et poivre . . . . .	17
2.13	Types des contours . . . . .	18
2.14	La convolution . . . . .	19
3.1	L'interface matlab . . . . .	22
4.1	Les étapes d'extraction de contours . . . . .	28
4.2	Le principe de la convolution . . . . .	28
4.3	Résultats sur image sans bruit . . . . .	33
4.4	Résultats sur image bruitée . . . . .	33
4.5	Résultat de détection par l'opérateur de Prewitt . . . . .	35
4.6	Illustration d'opérateur de roberts . . . . .	35
4.7	Résultat de détection par l'opérateur de roberts . . . . .	36
4.8	Avec le masque 3x3 . . . . .	38
4.9	Avec le masque 5x5 . . . . .	39
4.10	représentation des angles . . . . .	41
4.11	Avant et après suppression non-mximums . . . . .	41
4.12	Avant et après le seuillage par hystérésis . . . . .	42
4.13	Image noir et blanc . . . . .	43
4.14	Résultat de détection par l'opérateur de dériche . . . . .	46
5.1	Interface d'accueil de l'application . . . . .	48

5.2	Interface d'accueil de l'application . . . . .	49
5.3	figure agrandi . . . . .	50
5.4	menu de types des contours . . . . .	50
5.5	menu de types des bruit . . . . .	51
5.6	le seuillage . . . . .	51
5.7	paramètres de l'opérateur sobel . . . . .	52
5.8	paramètres de l'opérateur canny . . . . .	52
5.9	paramètres de l'opérateur deriche . . . . .	53

## Introduction générale

Lorsqu'on pose la question **qu'est-ce qu'une image ?** la réponse paraît simple, paraît évidente, mais en fait pas du tout. L'image est l'un des moyens de communication les plus influents et les plus utilisés de notre époque.

L'image est non seulement un moyen d'expression artistique, mais un outil réel de communication bien antérieur à l'écriture. Il existe des images fixes (dessins, peintures, gravures...) et des images animées (cinéma, image vidéo...) et chacun peut analyser ces images à sa manière pour dégager une impression et d'en extraire des informations précises.

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 comme titre d'exemple la reconnaissance, la classification et la détection de contour.

La détection de contours au sein d'une image est une caractéristique importante du processus de recherche d'image selon le contenu. Face à un nombre important de technique et filtre, les techniques de détection de contours analysent souvent une image dans globalité sans tenir compte des spécifiques des composantes de l'image. Pour mener à bien cette opération et remédier à cet état de fait, il est utile d'étudier dans un premier temps un ensemble de techniques et d'analyser leur performance et leur adaptabilité.

Dans ce mémoire, nous nous intéressons uniquement aux méthodes de détection de contours qui est une opération fondamentale en traitement d'images. Il consiste à extraire de l'image des primitives, de type contours .Ces primitives seront exploitées dans des nombreuses disciplines de traitement d'images. Donc la détection de contours n'est pas un objectif en soi, mais une étape située en amont de dispositifs d'interprétation d'images, de reconstruction tridimensionnelle ou codage etc.

Nous avons structuré notre mémoire comme suit :

- Chapitre I : introduction générale.
- Chapitre II : ce chapitre est dédié notions de base sur le traitement d'image et particulièrement sur les notions, contour, région et quelques opérations importantes (convolu-

lution et filtrage).

- Chapitre III : dans ce chapitre nous présentons les outils de développement utilisés pour la détection de contours(MATLAB).
- Chapitre IV : ce chapitre aborde la démarche de la détection de contours pour les images à niveaux de gris en présentant opérateurs classiques et adaptifs.
- Chapitre V : dans ce chapitre nous avons mise en place une application qui permet de détecter toutes les régions de l'image.

Nous clôturons notre travail par une conclusion générale sur notre projet.

# Chapitre 2

## Notions de Base

### 2.1 Introduction

Les images constituent l'un des moyens les plus importants qu'utilise les hommes pour communiquer, transmettre et livrer le savoir entre eux .C'est un moyen de communication universel dont la richesse du contenu permet aux êtres humains de tout âge et de toute culture de se comprendre. Les images sont les principales sources d'information dans de nombreuses applications, et constituent un moyen important et très utile dans tous les domaines.

### 2.2 Généralités sur le traitement d'images :

Le traitement d'images est un domaine très vaste qui connaît un développement depuis des années. Il désigne un ensemble de méthodes dont l'objectif soit de modifier l'image dans le but d'en extraire de l'information soit de transformer des images pour en améliorer l'apparence. Cette discipline se situe à la croisée des mathématiques appliquées et de l'informatique, c'est pourquoi nous nous intéresserons à la définition d'une image et ses notions.

#### 2.2.1 Définition :

Il est important de définir quelques notions élémentaires relatives à ce domaine, afin d'entamer les différentes techniques de traitement d'images concernant notre travail. Une image est définie comme étant la reproduction exacte ou la reproduction analogique d'une scène réelle. L'image contient en chaque

point, une intensité lumineuse perçu par un camera ou autre capteur (scanner, appareil photo, satellite...). Mathématiquement, l'image représente sous forme d'une fonction continue  $f$ , cette fonction s'appelée fonction d'image, de deux variables spatiales représentée par  $f(x,y)$  qui mesure la nuance du niveau de gris de l'image aux coordonnées (??, ??). On spécifie L'image numérique qui est une 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. Une image numérique est constituée de zones rectangulaires .Il existe plusieurs types de cette image parmi eux :

#### 2.2.1.1 Une image vectorielle :

Une image vectorielle nommée aussi image numérique composée par des objets géométriques individuels, des primitives géométriques (segment de droite, arcs cercle...) définis chacun par différentes attributs (forme, position, couleur). Ses avantages : les fichiers qui la composent sont petits, les redimensionnements sont faciles et sans perte de qualité. Ses inconvénients : une image vectorielle ne permet de représenter que des formes simples. Elle n'est pas donc utilisable pour la photographie notamment pour obtenir des photos réalistes.

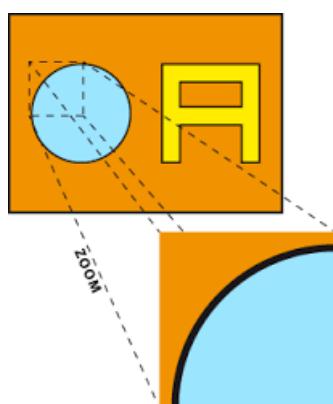


FIGURE 2.1 – Image vectorielle

#### **2.2.1.2 Image matricielle :**

Une image matricielle est une image constituée d'une matrice ou d'un tableau sous forme de grille où chaque case possède une couleur, il s'agit donc d'une juxtaposition de points colorés formant dans leur ensemble une image, ces derniers s'appellent des pixels.

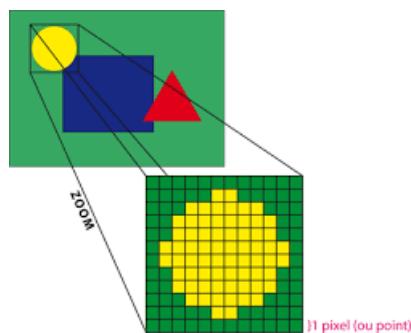


FIGURE 2.2 – Image matricielle

#### **2.2.1.3 Image à niveau de gris :**

Les images à niveau de gris renferment 256 teintes de gris, elles autorisent un dégradé de gris entre le noir et le blanc. Par convention la valeur 0 représente le noir et la valeur 255 le blanc. Le nombre 256 est lié à la quantification de l'image. En général, on code le niveau de gris sur un octet (8 bit). On peut aussi coder une image à niveau de gris sur 16 bits ou sur 2 bits : dans ce dernier cas le niveau de gris vaut 0 ou 1 : il s'agit alors d'une image binaire.



FIGURE 2.3 – Image à niveau de gris

#### 2.2.1.4 Image Binaire :

Une image binaire est une matrice rectangulaire dont les éléments valent 0 ou 1. Bien qu'il n'y ait que deux valeurs possibles. Lorsque l'on visualise une telle image, les 0 sont représentés par du noir et les 1 par du blanc.

1	1	1	1	1	1	1	1	1	1	1
1	0	0	0	1	1	0	0	0	1	
1	1	0	1	1	1	1	1	0	1	1
1	1	0	1	1	1	1	1	0	1	1
1	1	0	1	1	1	1	1	0	1	1
1	1	0	0	0	0	0	0	1	1	
1	1	0	1	1	1	1	1	0	1	1
1	1	0	1	1	1	1	1	0	1	1
1	1	0	1	1	1	1	1	0	1	1
1	0	0	0	1	1	0	0	0	1	
1	1	1	1	1	1	1	1	1	1	1

FIGURE 2.4 – Image binaire

#### 2.2.1.5 Image couleur :

On peut synthétiser les couleurs perceptibles par l'œil humain en superposant 3 couleurs de base « En général, on choisit le rouge, le vert, et le bleu, mais on peut réaliser la synthèse à l'aide d'autres ensembles de trois couleurs ». Pour représenter la couleur d'un pixel il faut donner 3 nombres, qui correspondant au dosage de 3 couleurs de base : rouge, vert et bleu. On peut ainsi représenter une image par trois matrices, chaque matrice correspondant à une couleur de base.



FIGURE 2.5 – Image couleur

## 2.2.2 Echantillonnage :

L'échantillonnage consiste à découper l'image suivant une grille régulière et à associer une intensité à chacune de ses cellules.

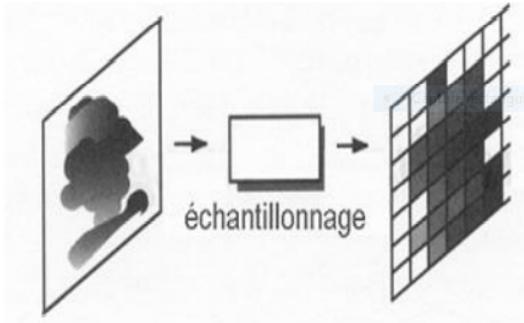


FIGURE 2.6 – Échantillonnage

## 2.2.3 Quantification :

Elle consiste à effectuer une valeur numérique à chaque échantillon. La quantification peut être monochrome ou couleur.

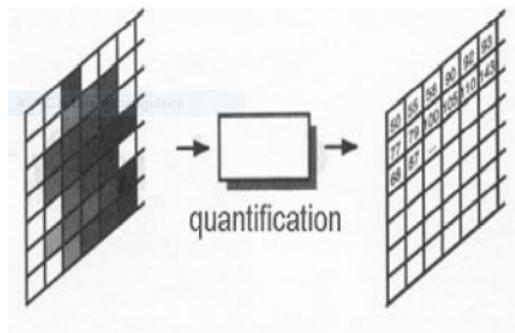


FIGURE 2.7 – Quantification

## 2.2.4 Caractéristiques d'image :

### 2.2.4.1 Pixels :

Chaque image numérique est constituée d'un ensemble de points appelés les pixels. Le pixel comme définition est l'unité de base permettant de mesurer la définition d'une image numérique matricielle. Le pixel est l'abréviation

élément d'image "Picture element" (en anglais). C'est la primitive de bas niveau la plus pauvre en information car ces seules propriétés sont la position dans la matrice image ( $N^{\circ}$  de ligne,  $N^{\circ}$  de colonne) et la valeur numérique indiquant sa couleur, ou son niveau de gris. Il peut être représenté en mémoire sur :

- Un bit (0 ou 1) pour les images monochromes : 0 pour le noir et 1 pour le blanc.
- Un octet, soit 256 niveaux de gris pour une image à niveau de gris : 0 noir et 255 blancs. La couleur du pixel est combinaison des trois nuances de chaque couleur trois octets pour une image couleur (RVB) :
  - 1 octet pour la couleur rouge (256 nuances de rouge).
  - 1 octet pour la couleur verte (256 nuances de vert).
  - 1 octet pour la couleur bleue (256 nuances de bleu).

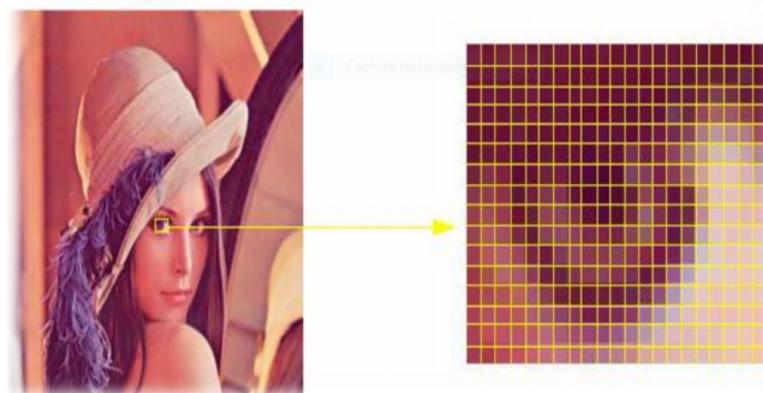


FIGURE 2.8 – Image numérique

#### 2.2.4.2 Histogramme :

Histogramme est un graphique statique permettant de représenter la distribution des intensités des pixels d'une image .C'est-à-dire le nombre de pixels pour chaque intensité lumineuse. L'histogramme permet de donner un grand nombre d'informations sur la distribution des niveaux de gris de l'image, ou

ce qu'on appelle « La dynamique de l'image ». Il permet de voir entre quelles bornes est repartie la majorité des niveaux de gris dans le cas d'une image trop claire ou trop foncée.

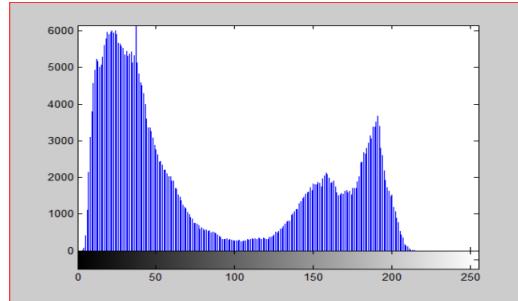


FIGURE 2.9 – Exemple d'histogramme

#### 2.2.4.3 Région :

Une région est un ensemble de pixels partageant des propriétés communes. C'est aussi l'ensemble de pixels connexes ayant un critère d'homogénéité ou de similarité donnée. Ce critère peut être, par exemple le niveau de gris, couleur, texture ....

#### 2.2.4.4 Bruit :

Le but définit les interférences d'un signal, ou les parties du signal sont déformées localement. Ainsi le bruit d'une image indique les pixels de l'image dont l'intensité est très distincte de celles des pixels voisins. Le bruit peut provenir de différentes causes :

- Qualité de l'échantillonnage.
- Capacité des capteurs ou une mauvaise utilisation de ces derniers.
- Environnement lors de l'acquisition.

#### 2.2.4.5 Types de bruit :

- *Bruit de Poisson* : Le bruit de Poisson ou le bruit de projectile est un type de bruit électronique qui se produit quand le nombre fini de particules qui portent l'énergie, telle que des électrons dans un

circuit électronique ou des photons dans un circuit optique, est assez petit pour provoquer des fluctuations statistiques discernables dans une mesure . Le nombre de photons mesurés par un photosite est aléatoire et dépend de l'illumination. Si un photosite mesure en moyenne photons, alors on est en présence d'un processus de Poisson de moyenne et de densité de probabilité.

- *Bruit d'amplificateur (bruit gaussien)* :Le modèle standard du bruit d'amplificateur est additif, gaussien, indépendant à chaque pixel et indépendant des caméras couleur du signal ou d'image d'intensité. Le bruit gaussien a une fonction de densité de probabilité de la distribution normale.

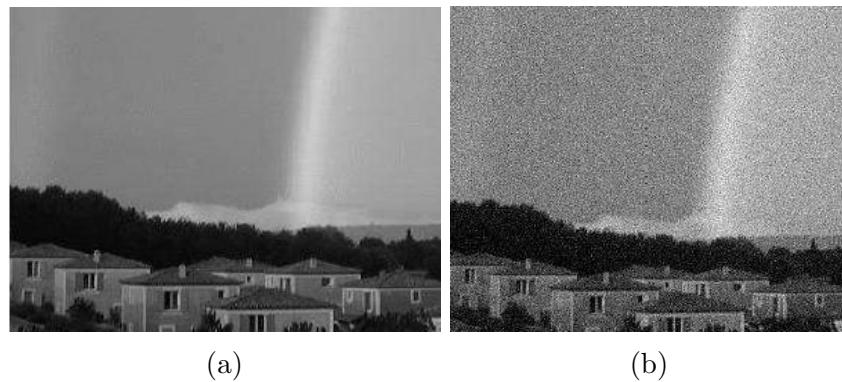


FIGURE 2.10 – (a)image originale (b)image bruitée par un bruit gaussien

- *Bruit de tache (Speckle)* :Le phénomène de Speckle ; ”chatoiement” dans les images radar, ou tavelure (en astronomie) ou granularité (en optique) et Speckle en anglais fut constaté la première fois en 1960, lors d'expérience d'illumination d'objets avec une source de lumière cohérente, le laser. La granularité détectée n'avait pas une relation simple avec les propriétés macroscopiques de l'objet. Tandis que le bruit gaussien peut être modelé par des valeurs aléatoires sur une image ; le bruit de Speckle peut être modelé par des valeurs aléatoires multipliées par les valeurs de Pixels, par conséquent il s'appelle également le bruit multiplicatif. Le bruit de Speckle est un problème important dans quelques applications de radar et d'échographie.



(a)

(b)

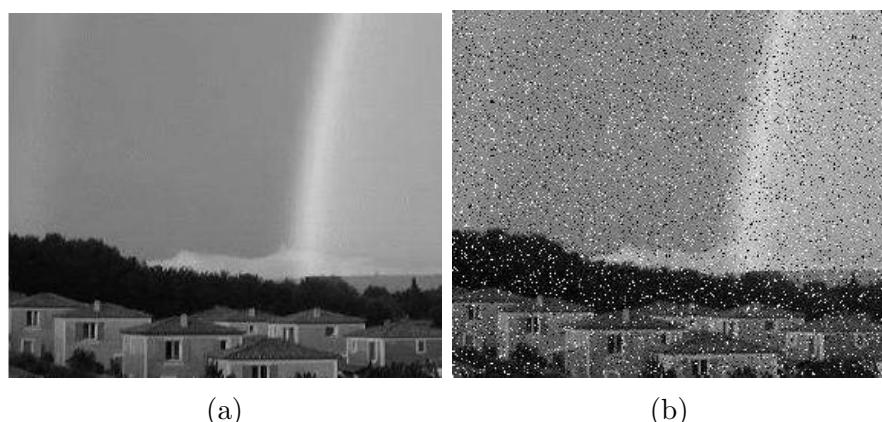
FIGURE 2.11 – (a)image originale (b)image bruitée par un bruit speckle

– *Bruit sel et poivre :*

Ce type contient des occurrences aléatoires des valeurs d'intensité de noir et de blanc, et souvent causé par le seuil de l'image bruitée . Le bruit de distribution du bruit « sel et poivre » peut être exprimé par :

$$p(x) = \begin{cases} p_a & x = a \\ p_b & x = b \\ 0 & \text{Ailleur} \end{cases}$$

Où  $P_a$ ,  $P_b$  sont les fonctions de densité de probabilités ,  $p(x)$  est la distribution du bruit « sel et poivre » dans l'image.



(a)

(b)

FIGURE 2.12 – (a)image originale (b)image bruitée par un bruit sel et poivre

#### 2.2.4.6 Contour :

Les contours est la frontière qui sépare deux objet dans une image et la partie la plus informative dans celle-ci. En particulier, les contours d'un objet permettent en générale de caractériser sa forme. Un contour correspond à une variation d'intensité ou une discontinuité entre les propriétés de deux ensembles de points. La détection de contours peut être réalisée grâce à des filtres dont les coefficients ont été soigneusement choisis. On distingue cependant les types de contours :

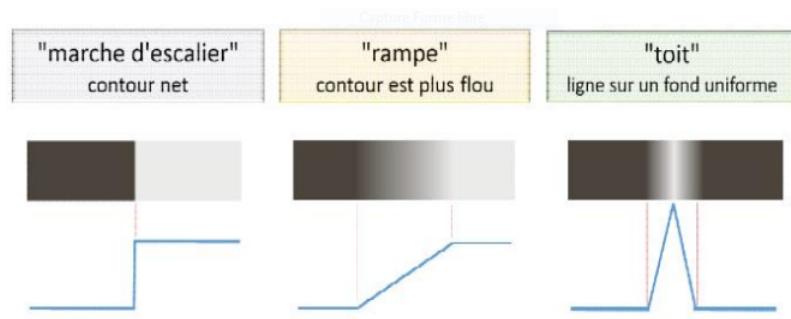


FIGURE 2.13 – Types des contours

#### 2.2.4.7 Convolution :

La convolution est courante en traitement d'images. Elle représente une classe d'opérations simple à programmer et efficace dans ses résultats. Elle consiste en une opération de multiplication de deux matrices de tailles différentes, mais de même dimensionnalité semblable, produisant une nouvelle matrice. Ce produit devra être noté que les opérations matricielles effectuées (les convolutions) ne sont pas des multiplications traditionnelles de matrices malgré le fait que ce soit noté par un « \* ». Le filtre parcourt toute la matrice principale de manière incrémentale et génère une nouvelle matrice constituée les résultats de la multiplication. Dans le traitement d'image, ceci est utilisé pour effectuer un flou gaussien ou détouurer les éléments d'une photo .

#### 2.2.5 Filtrage :

Les filtrages visent à modifier le contenu d'un pixel en prenant en compte une information locale. C'est-à-dire une information extraite du voisinage plus

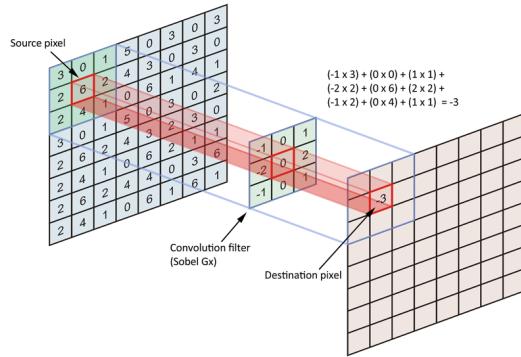


FIGURE 2.14 – La convolution

ou moins étendu du pixel. D'une façon générale, le filtrage est obtenu par convolution de l'image avec un noyau défini. Ce dernier peut être interprété comme une petite image ou vignette contenant un gabarit de transformation (linéaire ou non linéaire) et qu'applique sur chacun des pixels de l'image à filtrer pour créer une nouvelle image. Parmi les types de filtrage :

#### 2.2.5.1 Filtre linéaire :

Le filtrage linéaire est la convolution d'une image ( ??, ??) avec une fonction ( ??, ??) qui s'appelle réponse impulsionales du filtre. Ce filtre consiste à remplacer chaque niveau de gris par une combinaison linéaire des niveaux de gris des points voisins, les coefficients de cette combinaison linéaire sont définis par la réponse impulsionnelle du filtre. le traitement du signal au traitement des images numériques. Ceci est dû au fait qu'une image numutilisation des filtres linéaires provient de l'extension des méthodes mises au point pour le érique est considérée comme un signal bidimensionnel numérisé.

- **Filtre moyenneur.**
- **Filtre gaussien.**

#### 2.2.5.2 Filtres non-linéaires :

Ces opérateurs ont été développés pour pallier aux insuffisances des filtres linéaires : principalement la mauvaise conservation des contours. Ils ont le défaut d'infliger des déformations irréversibles à l'image et ils sont basés sur des bases mathématiques .On distingue les filtres ci-dessous :

- **Filtre médian.**

- **Filtre de Canny.**
- **Le filtre de Laplacien.**
- **Le filtre de gradient.**
- **Les filtres de Prewitt, Sobel, Robert, Freeman et Kirsch**

### 2.2.6 Conclusion

Conclusion Pour résumer, dans ce chapitre , nous avons illustré les généralités du traitement d'images. Nous avons présenté les concepts les plus importants liés à ce domaine. Les différentes définitions qui y sont développées sont celles des connaissances élémentaires de cette discipline mais quand bien même elles seraient essentielles pour l'initiation aux traitements approfondis des images.

# Chapitre 3

## Outils de développement

### 3.1 *Qui est-ce que Matlab :*

**Matlab** est l'abréviation de MATRIX LABORATORY, une traduction littérale qui nous amène à voir MATLAB comme un laboratoire pour manipuler des matrices. Nous reviendrons sur ce point, qui est un élément fondamental du langage MATLAB .Ce dernier est un langage de programmation relativement simple à assimiler en fait environnement ouvert et programmable qui permet un gain de productivité important. C'est un logiciel qui peut être utilisé pour le calcul ou le traitement de données, traçage de courbes et la résolution de systèmes et d'algorithmes de calculs numériques au sens large du terme. De plus, il comprend de nombreuses fonctions prédéfinies pour le calcul matriciel, mais pas seulement. Celles -ci dédiées à des domaines particuliers .Nous pouvons citer par exemple :

- le calcul de probabilités ou les statistiques ;
- le calcul intégral ou la dérivation ;
- le traitement du signal ;
- le traitement d'image ;

### 3.2 *Environnement Matlab :*

Pour lancer MATLAB, on peut utiliser le raccourci ou le menu **Démarrer tous les programmes**. Lorsque MATLAB est ouvert, une interface appelée

« bureau MATLAB » s'affiche. Par défaut cette interface est composée de quatre fenêtres, dont trois sont visibles et une cachée :



FIGURE 3.1 – L'interface matlab

- **menu** : regroupe des commandes de base de matlab à savoir :afficher, enregistrer, afficher les préférences, etc...
- **La zone de commandes** :  
C'est la fenêtre principale ; on y trouve l'invite de commandes (>>) ; c'est là que seront tapées les commandes exécutées par MATLAB. Une commande saisie se terminera par la touche « Entrer ».
- **la zone des variables** : Cette fenêtre liste de toutes les variables générées dans la session courante ; elle donne le type, et la taille des variables. On peut utiliser cette fenêtre pour tracer des graphiques, ou vérifier le contenu des variables.
- **Explorateur des fichiers** : Cette fenêtre présente le répertoire de travail ainsi que les fichiers en cours d'utilisation. On peut accéder rapidement à un répertoire ou un fichier par cette fenêtre.
- **L'historique des commandes** :  
L'historique des commandes garde en mémoire et affiche toutes commandes tapées, même celles des sessions précédentes de MATLAB. Ces

commandes peuvent être copiées dans fichier, ou dans la fenêtre de commandes pour être de nouveau exécutées.

### 3.3 Caractéristiques de Matlab :

**MATLAB** est un langage d'expressions qui sont introduites par l'utilisateur et interprétées par le système. Les commandes **Matlab** prennent la forme variable = expression ou simplement la forme expression. Cette dernière s'exécute de gauche à droite en considérant la priorité des opérations puis son évolution produit une matrice, qui est alors écrite sur l'écran et stockée pour une utilisation future. Le symbole ';' à la fin de la commande supprime l'affichage du résultat à l'écran. La déclaration des variables n'est pas importante, leur nature se déduisant automatiquement lors de l'affectation ce qui rend le programme écrit en **Matlab** plus facile à écrire et à lire par rapport à des programmes écrits en d'autre langage. Sur ce logiciel plusieurs fonctions et commandes peuvent être saisies dans un fichier qui sera enregistré avec l'extension .m .En éditant le nom de ce fichier dans la fenêtre déclarées de commande, l'ensemble des fonctions déclarées dans ce fichier seront exécutées.

### 3.4 Les fonctions de MATLAB utilisées pour un traitement d'image :

L'objectif de ce traitement d'images sous Matlab est de présenter la notion d'image et d'effectuer des opérations simples d'analyse d'images telles que :

- Lecture, écriture et affichage d'une image au niveau de gris.
- Changement d'espace couleur.
- Analyse et restauration.
- Détection de contours.

Les fonctions MATLAB utiles pour gérer les images sont les suivantes :

- **imread** : permet de lire une image
- **xlabel** : permet d'afficher le nom de l'axe des abscisses
- **ylabel** : permet d'afficher le nom de l'axe des ordonnes
- **imshow** : permet l'affichage d'image
- **title** : permet d'afficher un titre à la figure
- **figure** : permet de générer une fenêtre graphique permettant de visualiser les données (courbes, images)
- **imwrite** : permet d'enregistrer une image
- **plot** : permet de tracer une figure
- **subplot** : permet d'afficher plusieurs images dans une même fenêtre
- **imfinfo** : affiche les informations relatives à un fichier
- **im2double** : transforme les valeurs d'une image en valeurs réelles
- **rgb2gray** : transforme une image rgb en une image à niveau de gris
- **close** : permet de fermer la fenêtre active
- **close all** : permet de fermer toute les fenêtres
- **clear** : efface les variables mises en mémoire durant une session MATLAB
- **clc** : efface le contenu de la fenêtre de commande
- **pause** : permet de mettre en veille de commande de MATLAB
- **disp** : display permet l'affichage de texte
- **imrotate** : rotation d'une image
- **imnoise** : ajout de bruit

### **3.5 Détection de contours sous MATLAB :**

La détection de contour est une étape importante à de nombreuses applications de l'analyse et traitement d'images. Elle réduit de manière significative la qualité des données et élimine les informations. En pratique, il s'agit de reconnaître les zones de transition et de localiser au mieux la frontière entre les régions. Pour mieux comprendre cette notion il est possible de visualiser une image en 3D.

Un contour est donc défini comme une zone de l'image où l'intensité des pixels change subitement, cette interruption dans l'image est le passage d'un niveau de gris à un autre, de manière plus ou moins rapide.

### **3.6 Conclusion :**

Dans ce chapitre nous avons illustré l'environnement MATLAB qu'on va utiliser pour développer notre application .En bref MATLAB est un logiciel utile pour faire des calculs rapides peuvent se réaliser en une heure par contre aux autres programmes nécessitent une longue durée de programmation comme C et C++.

# Les méthodes d'extraction des contours

## 4.1 Introduction

La notion de contour étant reliée à celle de variation, il est évident qu'une telle définition nous amène tout naturellement vers une évaluation de la variation en chaque pixel .Une variation existera si le gradient est localement maximum ou si la dérivée seconde présente un passage par zéro. Les principaux algorithmes connus (Roberts, Prewitt, Pobel, Canny, Deriche,... ) se focalisent sur aspect du contour. Mais dans notre cas on va s'intéresser seulement sur les deux filtres : filtre de Sobel et filtre de Canny.

## 4.2 Méthodes de détection de contours

Un contour est un ensemble des points d'une image numérique qui correspond à un changement brutal de l'intensité lumineuse. Dans l'approche « Contour », on considère que les primitives à extraire sont les lignes de contrastes séparant des régions de niveaux de gris différents et relativement homogènes, ou bien des régions de texture différentes. En pratique, il s'agit de reconnaître les zones de transition et de localiser au mieux la frontière entre les régions. Le but de la détection de contours est de repérer les points d'une image numérique qui correspondent à un changement 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.

### 4.2.1 Méthodes dérivatives

Ces méthodes sont très faciles à l'implémentation ainsi que leur temps de calcul relativement court, et leur résultat satisfaisant pour des images non bruitées. Leur inconvénient est qu'elles sont très sensibles au bruit.

#### 4.2.1.1 Opérateur de gradient

Le gradient c'est une généralisation de la notion de dérivée pour une fonction à plusieurs variables. Le gradient d'une image défini l'orientation d'un contour au pixel considéré la direction (x,y), dirigé selon la direction de plus fort changement d'intensité (perpendiculaire au contour). On peut définir ces deux dérivées partielles :  $\frac{\partial f}{\partial x}$  et  $\frac{\partial f}{\partial y}$

À partir de ces deux valeurs on peut construire un vecteur, le gradient qu'on note :  $\vec{\text{grad}} f$  où  $\vec{\nabla}$  et désigné par :

$$\vec{\text{grad}} f = \vec{\nabla} f = \begin{pmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{pmatrix}$$

En chaque point de coordonnée (x,y), le vecteur gradient est caractérisé par :

1. **Module** : est liée à la quantité de variation locale de l'intensité, permet de quantifier l'importance du contour mis en évidence.

$$G = \|\vec{\nabla} f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

2. **La direction** : permet de déterminer l'arête présente dans l'image. En effet la direction du gradient est orthogonale à celle du contour.

$$\theta = \arctan\left(\frac{\frac{\partial f}{\partial y}}{\frac{\partial f}{\partial x}}\right)$$

Le gradient n'est autre qu'une dérivée vectorielle de l'image et permettent de détecter les contours du fait que les contours correspondent à des discontinuités d'ordre 0 de la fonction d'intensité. Cette approche permet une approximation des passages par zéro de la dérivée seconde de l'image qui détermine les transitions des intensités des pixels.

## 4.2.2 Méthodes déformables

### 4.2.2.1 Filtre de Sobel

Le filtre (l'opérateur) de Sobel est un outil utilisé en traitement d'image pour la détection de contours, réalisé par **Sobel-Feldman** en 1968. L'opérateur est basé sur le calcul de gradient de l'intensité de chaque pixel d'image, ceci indique la direction de la plus forte variation du clair au sombre (variation de contraste), mais aussi le taux de changement dans cette direction

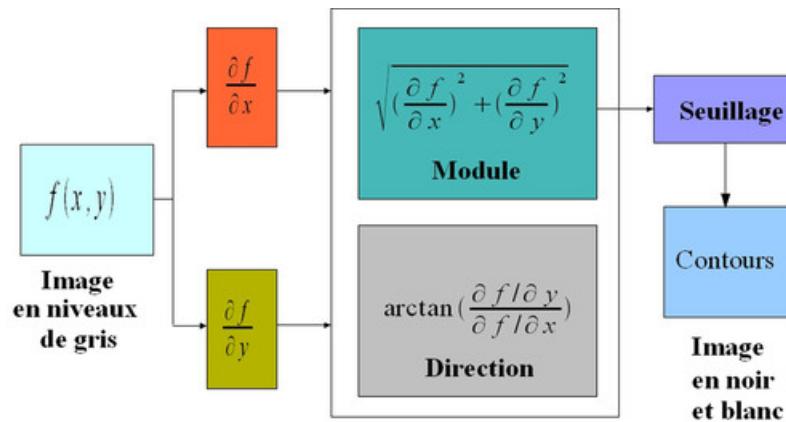


FIGURE 4.1 – Les étapes d'extraction de contours

L'opérateur utilise des matrices de convolution. La matrice est (généralement de taille 3x3 mais dans notre cas en va travailler avec 5x5) subit une convolution avec l'image pour calculer des approximations des deux dérivées.

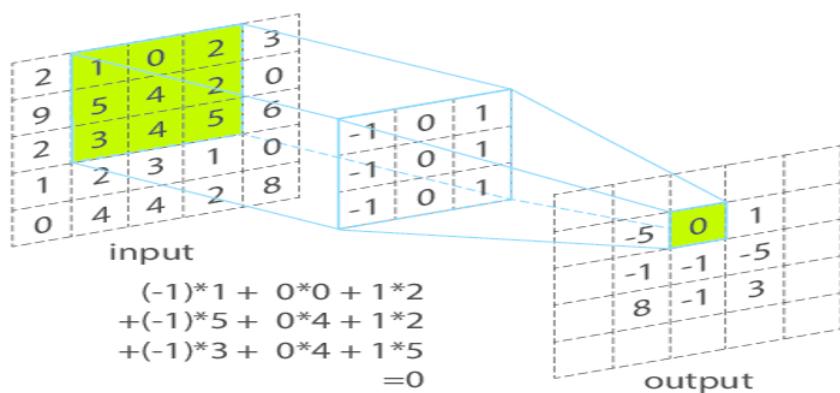


FIGURE 4.2 – Le principe de la convolution

Soit A la matrice correspond à 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 :

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

$$G_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{pmatrix} * A$$

En chaque point, les approximations des gradients horizontaux et verticaux peuvent être combinées afin d'obtenir la norme de gradient suivant :

$$\mathbf{G} = \sqrt{Gx^2 + Gy^2}$$

On peut aussi calculer la direction du gradient comme suit :

$$\theta = \arctan\left(\frac{Gx}{Gy}\right)$$

- **Détermination du masque de taille 3x3 :**

Pour déterminer le masque de Sobel de n'importe quelle taille, nous pouvons approximer le gradient en additionnant les projections de toutes les paires voisines-centre sur la direction de gradient.

En général on calcule le gradient dans une direction horizontale positive telle que son vecteur unitaire est  $h_x = (1, 0)$  par la relation suivante :

$$\frac{\partial f}{\partial x} = \frac{f(x + hx, y) - f(x, y)}{hx} = f(x + 1, y) - f(x, y)$$

Puis on calcule dans une direction verticale positive avec le vecteur unitaire  $h_y = (0, 1)$  :

$$\frac{\partial f}{\partial y} = \frac{f(x, y + hy) - f(x, y)}{hy} = f(x, y + 1) - f(x, y)$$

Pour particulier le calcul du gradient pour le masque de Sobel (3x3) soit la matrice suivante :

$$A = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

On va calculer le gradient pour chaque pixel de la dans les directions différentes par : **la valeur d'un pixel moins la valeur au centre e, divisée par la distance entre eux suivant le vecteur unitaire reliant les deux pixel.**

On détermine valeur par valeur suivant l'axe horizontale( la projection suivant l'axe orizontale par la multiplication par  $h_x$ ) :

$$a : \frac{a - e}{ae} * \frac{\vec{ae}}{\|\vec{ae}\|} * h_x = \frac{a - e}{\sqrt{2}} * \frac{(-1, -1)}{\sqrt{2}} * (1, 0) = -\frac{a - e}{2}$$

$$b : \frac{b - e}{be} * \frac{\vec{be}}{\|\vec{be}\|} * h_x = \frac{b - e}{1} * \frac{(0, -1)}{\sqrt{1}} * (1, 0) = 0$$

$$c : \frac{c - e}{ae} * \frac{\vec{ce}}{\|\vec{ce}\|} * h_x = \frac{c - e}{\sqrt{2}} * \frac{(1, -1)}{\sqrt{2}} * (1, 0) = \frac{c - e}{2}$$

$$d : \frac{d - e}{de} * \frac{\vec{de}}{\|\vec{de}\|} * h_x = \frac{d - e}{1} * \frac{(-1, 0)}{1} * (1, 0) = -(d - e)$$

$$f : \frac{f - e}{fe} * \frac{\vec{fe}}{\|\vec{fe}\|} * h_x = \frac{f - e}{1} * \frac{(1, 0)}{1} * (1, 0) = f - e$$

$$g : \frac{g - e}{ge} * \frac{\vec{ge}}{\|\vec{ge}\|} * h_x = \frac{g - e}{\sqrt{2}} * \frac{(-1, 1)}{\sqrt{2}} * (1, 0) = -\frac{(g - e)}{2}$$

$$h : \frac{h - e}{he} * \frac{\vec{he}}{\|\vec{he}\|} * h_x = \frac{h - e}{1} * \frac{(0, 1)}{1} * (1, 0) = 0$$

$$i : \frac{i - e}{ie} * \frac{\vec{ie}}{\|\vec{ie}\|} * h_x = \frac{i - e}{\sqrt{2}} * \frac{(1, 1)}{\sqrt{2}} * (1, 0) = \frac{i - e}{2}$$

Après avoir calculé les contributions. On fait la somme de toutes les contributions pour obtenir le gradient horizontal.

$$G'_x = \frac{c-g}{2} + \frac{i-a}{2} + (f-d) + 0$$

On multiplie l'équation par 2 pour enlever  $\frac{1}{2}$ , c'est la normalisation qui ramène les valeurs normales et génère un lissage fort, on obtient donc :

$$G_X = 2G'_x = c - g + i - a + 3f - 2d = (i + 2f + c) - (g + 2d + a)$$

D'où le masque de taille 3x3 suivant l'axe horizontal est correspond à :

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

On refait le même calcul pour l'axe verticale avec le vecteur unitaire  $h_y = (0, 1)$  on trouve le résultat suivant :

$$G_y = (g + 2h + i) - (a + 2b + c)$$

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

### • Généralisation au masque de taille 5x5

Suivant la même procédure que précédemment on trouve les résultats correspondant au masque de taille 5x5 par rapport à l'axe horizontal pour la matrice ci-dessous :

$$\begin{pmatrix} a & b & c & d & e \\ f & g & h & i & j \\ k & l & m & n & o \\ p & q & r & s & t \\ u & v & w & x & y \end{pmatrix}$$

$$a : \frac{a-m}{am} * \frac{a\vec{m}}{\|a\vec{m}\|} * h_x = \frac{a-m}{\sqrt{8}} * \frac{(-2, -2)}{\sqrt{8}} * (1, 0) = -\frac{a-m}{4}$$

$$b : \frac{b-m}{be} * \frac{b\vec{m}}{\|b\vec{m}\|} * h_x = \frac{b-m}{\sqrt{5}} * \frac{(-1, -2)}{\sqrt{5}} * (1, 0) = -\frac{(b-m)}{5}$$

$$f : \frac{f-m}{fm} * \frac{\vec{fm}}{\|\vec{fm}\|} * h_x = \frac{f-m}{\sqrt{5}} * \frac{(-2, -1)}{\sqrt{5}} * (1, 0) = -2 \frac{(f-m)}{5}$$

$$g : \frac{g-m}{gm} * \frac{\vec{gm}}{\|\vec{gm}\|} * h_x = \frac{g-m}{\sqrt{2}} * \frac{(-1, -1)}{\sqrt{2}} * (1, 0) = -\frac{g-m}{2}$$

$$k : \frac{k-m}{km} * \frac{\vec{km}}{\|\vec{km}\|} * h_x = \frac{k-m}{2} * \frac{(-2, 0)}{2} * (1, 0) = -\frac{k-m}{2}$$

$$l : \frac{l-m}{lm} * \frac{\vec{lm}}{\|\vec{lm}\|} * h_x = \frac{l-m}{1} * \frac{(-1, 1)}{1} * (1, 0) = -(l-m)$$

De même manière on termine le calcul pour les autres valeurs et on trouve :

$$G' = [\frac{1}{2}(i - r - g + t + o - k) + (n - 1) + \frac{1}{5}(d - w - b + y) + \frac{1}{4}(e - v - a + z) + \frac{2}{5}(j - p - f - v)]$$

On fait la normalisation en multipliant l'équation par 20 pour rendre les valeurs normales, on obtient donc :

$$G' = [10(i - r - g + t + o - k) + 20(n - 1) + 4(d - w - b + y) + 5(e - v - a + z) + 8(j - p - f - v)]$$

D'où le masque suivant l'axe horizontal est :

$$G_x = \begin{pmatrix} -5 & -4 & 0 & 4 & 5 \\ +8 & -10 & 0 & 10 & 8 \\ +10 & -20 & 0 & 20 & 10 \\ -8 & -10 & 0 & 10 & 8 \\ -5 & -4 & 0 & 4 & 5 \end{pmatrix}$$

On refait le même calcul pour l'axe verticale avec le vecteur unitaire  $h_y = (0, 1)$  on trouvera les résultats suivants :

$$G_y = \begin{pmatrix} -5 & +8 & +10 & -8 & -5 \\ -4 & -10 & -20 & -10 & -4 \\ 0 & 0 & 0 & 0 & 0 \\ +4 & +10 & +20 & +10 & +4 \\ +5 & +8 & +10 & +8 & +5 \end{pmatrix}$$

#### 4.2.2.2 Implémentation avec MATLAB

Après avoir implémenter les résultats précédents on trouve les résultats suivant :

- image sans bruit :



FIGURE 4.3 – Résultats sur image sans bruit

- image avec bruit (bruit gaussien) :



(a) Image originale

(b) Avec le masque 3x3

(c) Avec le masque 5x5

FIGURE 4.4 – Résultats sur image bruitée

#### 4.2.2.3 Comparaison des résultats

Les résultats montrent que les contours de l'image détectés à l'aide de masques 5x5 apparaissent plus épais que celui avec le masque 3x3, aussi on constate que lors de l'application d'un masque 3x3 les contours d'image sont interrompus tandis que dans un masque 5x5 les contours

sont plus continu.

On constate aussi que le résultat d ?application du filtre de sobel est bon :

- pour un masque 3x3 appliqué à une image non bruitée.
- pour un masque 5x5 appliqué à une image bruitée.

#### 4.2.2.4 filtre de prewitt

le filtre de prewitt est utilisé en traitement d’image pour la détection de contour , et aussi c’est un opérateur de dérivation. Ce filtre consiste en un pair de masque de convolution 3X3 De type ci dessous :

$$\nabla_x = \begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix} \quad \nabla_y = \begin{pmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix}$$

Ces masques sont conçus pour répondre maximalement aux contours horizontaux et verticaux. L’image est convoluer avec les masques  $\nabla_x$  et  $\nabla_y$  ,tel que ces masques Rassembler respectivement les composantes horizontale et verticale du gradient.

Il est ensuite possible de calculer la norme et la direction du gradient en chaque point a partir des deux composantes  $\nabla_x$  et  $\nabla_y$  , la norme du gradient en chaque pixel est donne par la relation suivante :

$$||\nabla|| = \sqrt{\nabla_x^2 + \nabla_y^2}$$

L’angle d’orientation du gradient est fourni par la relation :

$$\theta = \arctan\left(\frac{\nabla_y}{\nabla_x}\right) - \frac{3\pi}{4}$$



FIGURE 4.5 – Résultat de détection par l'opérateur de Prewitt

#### 4.2.2.5 filtre de Roberts

Le filtre de Roberts permet de calculer le gradient d'une image en suivant les diagonales grâce aux deux matrices suivantes :

$$\text{la matrice Sud-Ouest, Nord-Est : } \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

$$\text{la matrice Nord-Ouest, Sud-Est : } \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$

La technique ici sera d'appliquer les deux matrices sur la même matrice de base de façon parallèle afin d'obtenir deux matrices filtrées séparément et de les fusionner par la suite :

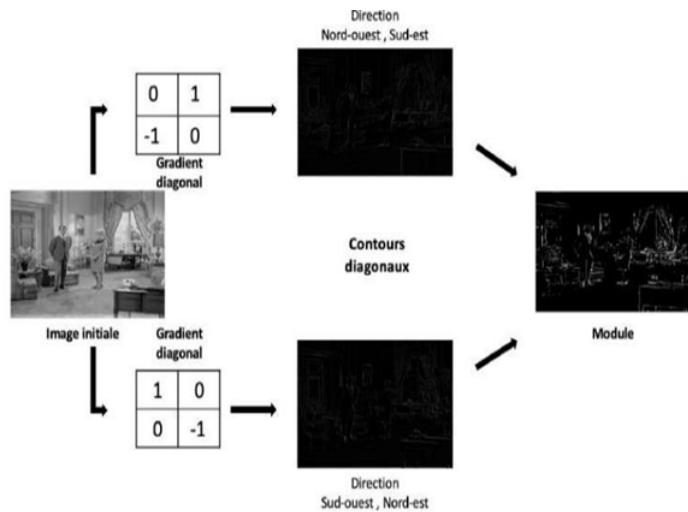


FIGURE 4.6 – Illustration d'opérateur de roberts

L'application des opérations de convolutions à une image permet de calculer les approximations du gradient  $G_x$  et  $G_y$ . Ceci nous permet de calculer la norme du gradient :

$$\mathbf{G} = \sqrt{Gx^2 + Gy^2}$$



FIGURE 4.7 – Résultat de détection par l'opérateur de roberts

### 4.2.3 Méthodes analytique

#### 4.2.3.1 Filtre de Canny

Le filtre de Canny permet de déterminer la détection des contours. L'algorithme a été conçu par John Canny en 1986 pour être optimal sur trois contraints suivant :

- **Bonne détection** : Ce critère revient à chercher un opérateur de détection et tel que le rapport signal sur le bruit soit maximum. C'est-à-dire faible taux d'erreur dans la signalisation des contours.
- **Bonne localisation** : Ce critère corresponds la minimisation de variance de la position des passages par zéro et revient à maximiser la localisation.
- **Unicité de la réponse** : Ce critère correspond à la limitation du nombre de maxima locaux. Il essaye de donner une seule réponse par contour et pas de faux positifs.

#### 4.2.3.2 Etapes d'opérateur de Canny :

- **Réduction du bruit** : C'est la première étape de l'algorithme qui concerne la réduction du bruit de l'image originale avant d'en détecter les contours, ceci permet d'éliminer les pixels isolés qui pourraient induire de fortes réponses lors du calcul du gradient. Pour faire ceci on utilise un filtre Gaussien.

Un filtre gaussien est un filtre linéaire dont les éléments du noyau

de convolution sont déterminés selon la densité d'une loi gaussienne centrée à 2 dimensions :

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp -\frac{x^2 + y^2}{2\sigma^2}$$

avec  $\sigma$  son écart-type.

On calcule le masque de gaussien M par la formule précédente en considérant  $\sigma = 0.8$

\* Pour la taille 3x3 :

$$G = \begin{pmatrix} G(-1, -1) & G(0, -1) & G(1, -1) \\ G(-1, 0) & G(0, 0) & G(1, 0) \\ G(-1, 1) & G(0, 1) & G(1, 1) \end{pmatrix}$$

calculons :

$$G(-1, -1) = \frac{1}{2\pi*0.8} \exp -\frac{(-1)^2 + (-1)^2}{2 * (0.8)^2} \approx 1$$

$$G(-1, 0) = \frac{1}{2\pi*0.8} \exp -\frac{(-1)^2 + (0)^2}{2 * (0.8)^2} \approx 2$$

$$G(0, 0) = \frac{1}{2\pi*0.8} \exp -\frac{(0)^2 + (0)^2}{2 * (0.8)^2} \approx 4$$

De même façon on calcule les autres éléments on trouve que :

$$M = \begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$$

Pour normaliser ce masque on le multiplie par  $\frac{1}{16}$  avec :

$$N = G(-1, -1) + G(-1, 0) + G(-1, 1) + G(0, -1) + G(0, 0) + G(0, 1) + G(1, -1) + G(1, 0) + G(1, 1)$$

dans notre cas N=16, alors le résultat final est :

$$M = \frac{1}{16} \begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$$

\* Pour la taille 5x5 :

la même procédure que la taille 3x3 :

$$M = \begin{pmatrix} G(-2, -2) & G(-1, -2) & G(0, -2) & G(1, -2) & G(2, -2) \\ G(-2, -1) & G(-1, -1) & G(0, -1) & G(1, -1) & G(2, -1) \\ G(-2, 0) & G(-1, 0) & G(0, 0) & G(1, 0) & G(2, 0) \\ G(-2, 1) & G(-1, 1) & G(0, 1) & G(1, 1) & G(2, 1) \\ G(-2, 2) & G(-1, 2) & G(0, 2) & G(1, 2) & G(2, 2) \end{pmatrix}$$

alors le résultat final est :

$$M = \frac{1}{1444} \begin{pmatrix} 1 & 7 & 16 & 7 & 1 \\ 7 & 75 & 164 & 75 & 7 \\ 16 & 164 & 359 & 164 & 16 \\ 7 & 75 & 164 & 75 & 7 \\ 1 & 7 & 16 & 7 & 1 \end{pmatrix}$$

Après avoir implémenter les résultats précédents on trouve les résultats suivant :

· Avec le masque de taille 3x3 :



(a) Image sans bruit

(b) Image bruitée

FIGURE 4.8 – Avec le masque 3x3



(a) Image sans bruit

(b) Image bruitée

FIGURE 4.9 – Avec le masque 5x5

- **Avec le masque de taille 5x5 :**

- \* **Gradient d'intensité :**

On calcule les dérivées premières de l'image lissée préalablement par une Gaussienne

$$f_x = \frac{\partial(f(x, y) * G(x, y))}{\partial x} = f(x, y) * \frac{\partial G(x, y)}{\partial x} = \frac{-x}{\sigma^2} f(x, y) * G(x, y)$$

$$f_y = \frac{\partial(f(x, y) * G(x, y))}{\partial y} = f(x, y) * \frac{\partial G(x, y)}{\partial y} = \frac{-y}{\sigma^2} f(x, y) * G(x, y)$$

Après le filtrage gaussien, on applique un gradient qui retourne l'intensité des contours, l'opérateur utilisé permet de calculer le gradient suivant les directions x et y, il est composé de deux masques de convolution : Un de dimension 3x1 et un autre de

$$1 \times 3 ; G_x = (-1, 0, 1), G_y = \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix}$$

$$|G| = |G_x| + |G_y|$$

Et pour déterminer la direction des contours : on la détermine par la relation suivante :  $\theta = \arctan(\frac{G_x}{G_y})$

Et après ce calcul on trouve une carte des gradients des intensités en chaque point de l'image accompagnée des directions des

contours.

\* **Suppression des non-maximums :**

La suppression des non-maximums est une étape assez rapide. Elle s'agit de supprimer la norme du gradient de toutes les valeurs «faibles» obtenue dans la carte des gradients. Cette dernière apporte une intensité en chaque point de l'image, une forte intensité indique une forte probabilité de présence d'un contour.

Les points qui correspondant à des maximums locaux sont considérés comme correspondant à des contours, et sont réservés pour la prochaine étape de la détection.

Les étapes de cette suppression pour chaque pixel sont les suivantes :

- Trouver les voisinages du pixel en suivant l'angle du gradient et faire un ajustement des directions : direction x et direction y ;

Si l'angle de gradient est compris entre 157,5 et 202,5 on lui remplace par 0 , Si l'angle est de 90 degrés, les pixels voisins seront le pixel juste au-dessus et juste en dessous du pixel courant. Ces pixels auront les coordonnées (x , y-1) et (x ,y+1). Trouver les voisins du pixel en suivant l'angle du gradient :  $\theta(i, j)$  P c'est un point de l'image et chaque pixel prend l'une des valeurs suivantes qui correspond à sa direction :

- $-22.5 < \theta(p) \leq 22.5^\circ$  : alors dans ce cas  $\theta$  prendre valeur  $0^\circ$ .
- $22.5 < \theta(p) \leq 67.5^\circ$  : alors  $\theta$  prendre la valeur  $45^\circ$ .
- $67.5 < \theta(p) \leq 112.5^\circ$  : Alors  $\theta$  prendre la valeur  $90^\circ$ .
- $-90 < \theta(p) \leq -67.5^\circ$  : alors  $\theta$  prendre la valeur  $90^\circ$ .
- $-67.5 < \theta(p) \leq -22.5^\circ$  : alors  $\theta$  prendre la valeur  $135^\circ$ .
- Vérifier la valeur des voisins : si la valeur de l'un des voisins est plus grande que la valeur du pixel courant, supprimer le pixel courant et lui affecter la valeur 0, sinon on le garde

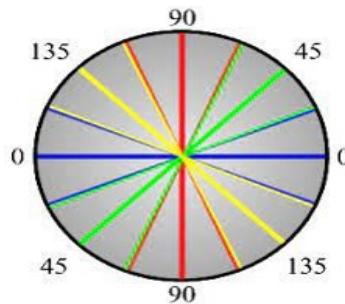


FIGURE 4.10 – représentation des angles

à sa valeur d'origine. Le résultat obtenu avant et après la suppression des non maximums :

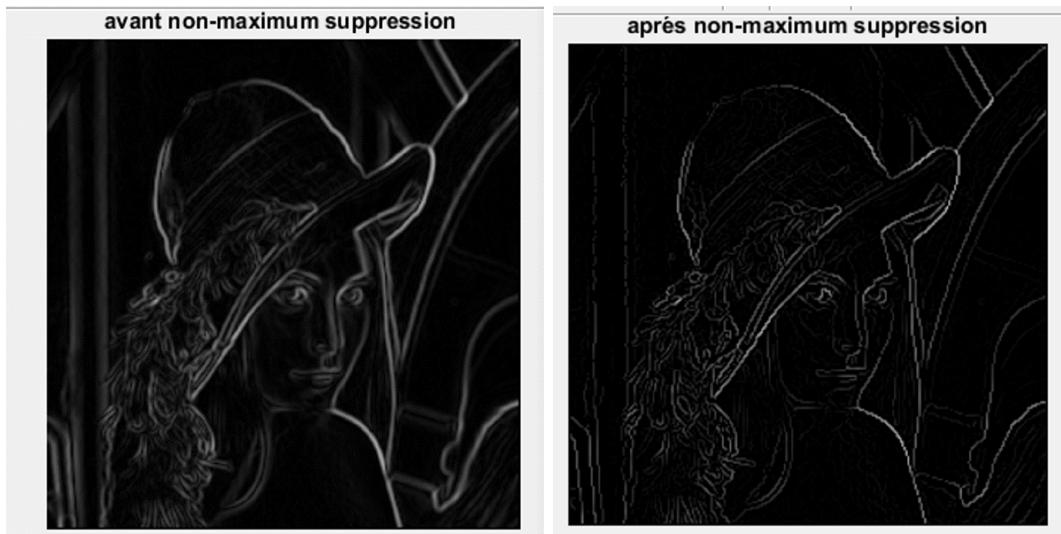


FIGURE 4.11 – Avant et après suppression non-mximum

- \* **Seuillage des contours (seuillage par hystérésis)** : Le seuillage par hystérésis est une méthode de détection de contours basée sur le gradient de l'image. Les contours donnés par les fortes valeurs du gradient sont souvent étalés.  
A cet effet, les pixels de l'image de gradient sont filtrés grâce à deux seuils distincts,  $T_1$  et  $T_2$  tel que  $T_1 < T_2$ .  
En dessous du seuil bas  $T_1$ , on considère qu'il n'y a pas de contours.  
Au-dessus du seuil haut  $T_2$ , on décide qu'il y a contour : ces

contours sont ensuite complétés par les pixels compris entre T1 et T2 si et seulement s'ils sont connexes entre eux. En résumé, les étapes de l'algorithme sont les suivantes :

- on effectue un calcul de gradient sur l'image à niveau de gris.
- tous les pixels en dessous de T1 sont mis à zéro.
- tous les pixels au-dessus de T2 sont conservés.
- on ne garde les pixels dont le niveau est entre T1 et T2 que s'ils sont connexes à au moins un pixel de niveau supérieur à T2.

Les résultats obtenus avant et après seuillage par hystérésis :



FIGURE 4.12 – Avant et après le seuillage par hystérésis

D'après les résultats précédents on constate que le filtre de Canny détecte les contours plus précisément que celui de Sobel car il l'améliore en considérant aussi la direction du gradient , l'ajout de la suppression des contours non maximum et le seuillage hystérésis

#### 4.2.3.3 Exemple d'application :

\* suppression de non maximum :

Soit la matrice extraite de l'image :



FIGURE 4.13 – Image noir et blanc

$$\mathbf{A} : \begin{pmatrix} 250 & 1 & 1 \\ 240 & 1 & 2 \\ 230 & 0 & 3 \end{pmatrix}$$

La colonne au milieu de la matrice est correspond au contour.  
On va vérifier si le pixel au centre de A est un maximum (vrai contour ) au pas.

Après le calcul le gradient horizontal et le gradient vertical on obtient le résultat :

$$G_x = \begin{pmatrix} -3 & 736 & 5.83 \\ -3 & 952 & 4.24 \\ -1 & 632 & 5.09 \end{pmatrix} \quad G_y = \begin{pmatrix} -481 & -244 & -5 \\ 41.10 & 20 & -3 \\ 481 & 244 & 1 \end{pmatrix}$$

On calcule la norme de gradient par la formule :

$$G = \sqrt{G_x^2 + G_y^2}$$

On obtient

$$G = \begin{pmatrix} 481 & 775.39 & 5.83 \\ 41.10 & 952.21 & 4.24 \\ 481 & 733.75 & 5.09 \end{pmatrix}$$

On peut aussi calculer la direction du gradient du pixel au centre comme suit :

$$\theta = \arctan\left(\frac{G_x}{G_y}\right) = \arctan\left(\frac{30}{952}\right) = 1.20 = 1.20 * \frac{180}{\pi} = 68.95^\circ$$

Puisque  $67.5^\circ < \theta = 68.95^\circ < 112.5^\circ$  donc on lui fait l'ajustement à  $90^\circ$ .

D'où les pixels voisins de pixel au centre sont juste au-dessus et juste en dessous (1 et 0) :

$$A = \begin{pmatrix} 250 & 1 & 1 \\ 240 & 1 & 2 \\ 230 & 0 & 3 \end{pmatrix} \quad G = \begin{pmatrix} 481 & 775.39 & 5.83 \\ 41.10 & 952.21 & 4.24 \\ 481 & 733.75 & 5.09 \end{pmatrix}$$

On compare les valeurs obtenus par la norme de gradient de pixel au centre avec celles des pixels voisins :

D'après la matrice G puisque :  $733.75 \leq 952.21$  et  $775.39 \leq 952.21$

D'où le pixel au centre est un maximum de gradient.  
Donc on lui garde sa valeur à 1.

\* seuillage par hystérésis :

On considère deux seuils ; seuil bas ( $T_b = 0.6$ ) et seuil haut ( $T_h = 0.72$ ) .

Pour appliquer ce seuillage à la matrice G on fait l'ajustement des seuils comme suit :

$$t_b = 0.6 * \max(G) = 0.6 * 952.25 = 571.32$$

$$t_h = 0.6 * \max(G) = 0.72 * 952.25 = 685.59$$

Avec  $\max(G)$  est la valeur maximal dans la matrice G

On constate que  $t_b < t_h < 952.21$ , donc le pixel au centre doit être conservé(vrai contour).

#### 4.2.3.4 Les avantages et les inconvénients de filtre de Canny :

- \* **Avantages :**

- Bonne localisation des contours.
- Une meilleure détection spécialement dans les images bruitées.

- \* **Inconvénients :**

- Les calculs complexes.
- Détecteur long par rapport à Sobel.

#### 4.2.3.5 DéTECTEUR DE DÉRICHÉ

Le détecteur de Deriche est un opérateur de détection de bord développé par Rachid Deriche en 1987. Au filtre de Canny, on préfère souvent le détecteur de Deriche, qui répond exactement aux mes critères de qualité que celui de Canny, mais qui possède une réponse impulsionale infinie. Deriche montre que le filtre défini par l'équation :

$$f_{der1}(x) = -\exp^{-\alpha|x|} \cdot \sin(wx)$$

( avec  $\alpha$  et  $w$  Positifs ) est plus performant que celui défini par le filtre de canny  $f_{can}(x) = x \cdot \exp^{-\alpha^2 x^2}$

Tel que  $\alpha$  le paramètre de Deriche :  $\alpha = \frac{\sqrt{\pi}}{\sigma}$  Les performances du détecteur de contours tendant vers un maximum lorsque le paramètre  $w$  tend vers zéro, Deriche propose d'utiliser le filtre dont la réponse impulsionale est donnée par :

$$f_{der2}(x) = -x \cdot \exp^{-\alpha|x|}$$

#### 4.2.3.6 Conclusion :

Ce chapitre nous a mené à montrer les performances des deux différentes techniques de détection de contours qui sont filtre de Canny et de filtre



FIGURE 4.14 – Résultat de détection par l'opérateur de dériche

de Sobel. Toutefois ces techniques ne donnent pas les mêmes résultats et certaines sont plus adaptées à un tel type d'images plutôt qu'à un autre. Généralement la détection des contours dans une image réduit de manière significative la quantité de données en conservant des informations qu'on peut juger plus pertinentes. Il existe un grand nombre de méthodes de détection des contours de l'image mais la plupart d'entre elles peuvent être regroupées en deux catégories. La première recherche les extremaums de la dérivée première. La seconde recherche les annulations du dérivé second.

# Application

## 5.1 Introduction :

Afin de rendre l'utilisation des fonctions de traitement d'image plus facile, nous avons choisi de créer une interface graphique interactive MATLAB, navigable à la souris ou au clavier. Celle-ci est basée sur GUIDE, qui est l'outil de développement d'interfaces utilisateurs graphiques inclus avec MATLAB. Il offre l'avantage de pouvoir facilement intégrer des fonctions MATLAB.

## 5.2 Les fonctionnalités de l'application :

L'interface de l'application se compose d'une fenêtre principale rassemblant les fonctionnalités suivant :

- \* Extraire les contours des images par l'utilisation de différentes méthodes.
- \* Paramétrage des opérations selon les résultats attendus.
- \* Applications de seuillage par la méthode fixe.
- \* Application de bruit sur l'image.
- \* Affichage de l'histogramme.
- \* Affichage de l'image original avec des luminosités différentes.

### 5.3 Réalisation de l'interface :

Cette Interface est composée de trois parties principales, deux zones pour affiché les images (avant et après le traitement) , un groupe des boutons qui manipulent les données (l'affichage et la modification du l'image qu'on désire traiter) et une troisième partie contient ensemble des boutons qui seront utilisées pour fixer les paramétrages selon chaque opérateur .



FIGURE 5.1 – Interface d'accueil de l'application

### 5.4 Description de l'interface :

\* Outils généraux

- **Axes** :sont des zones utilisés pour afficher les images(avant et après le traitement).
- **Bouton Insérer l'image** :charger l'image à traiter et l'affiche dans la premiere zone (axe 1).

- **Bouton Appliquer** :lancer l'exécution après avoir fixer les paramètres d'opérateur choisi et afficher le résultat final dans la deuxième zone.
- **Bouton Histogramme** : pour afficher l 'histogramme de l'image choisi.
- **Bouton image originale** permet d'afficher un nouveau glissière pour modifier la luminosité et aussi un champ text dans lequel apparait la valeur de luminosité choisie et aussi un bouton annuler apparaître comme le montre le figure ci-dessous.

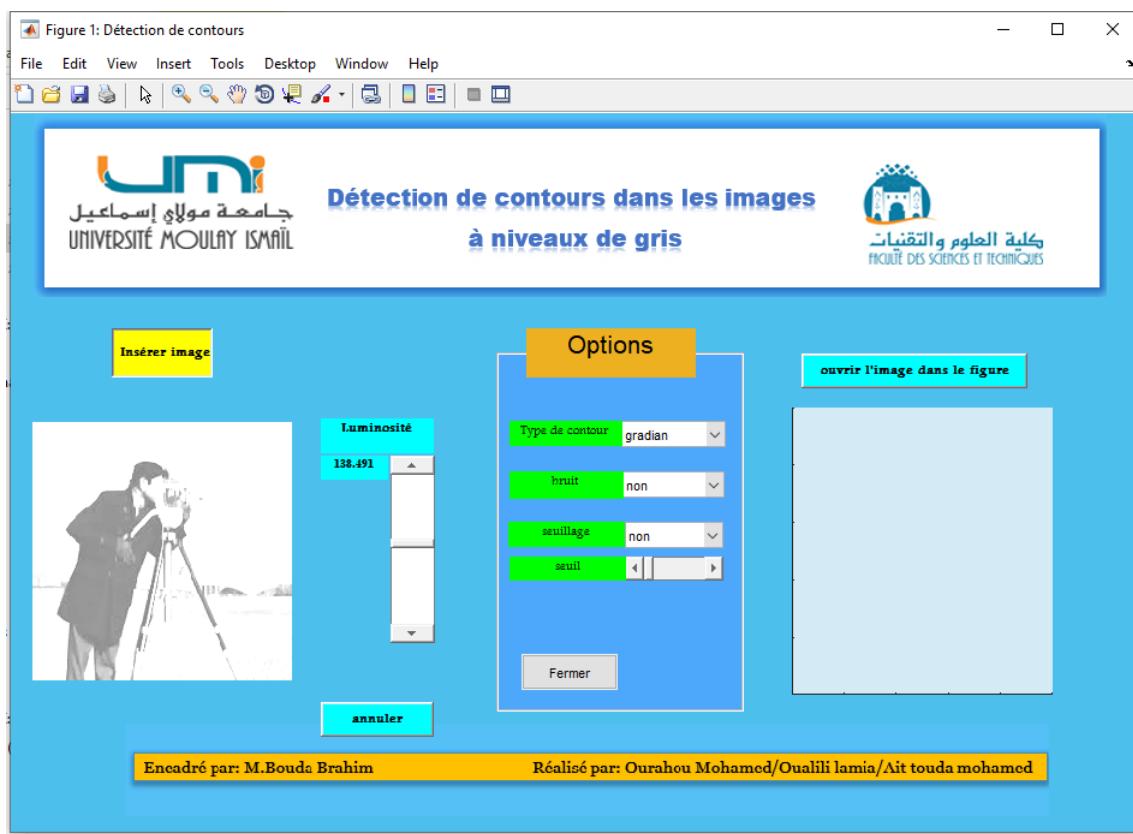


FIGURE 5.2 – Interface d'accueil de l'application

- **Bouton supprimer** :libère les zones des images.
- **Bouton ouvrir l'image dans le figure** : agrandir l'image traitée dans un grand figure.

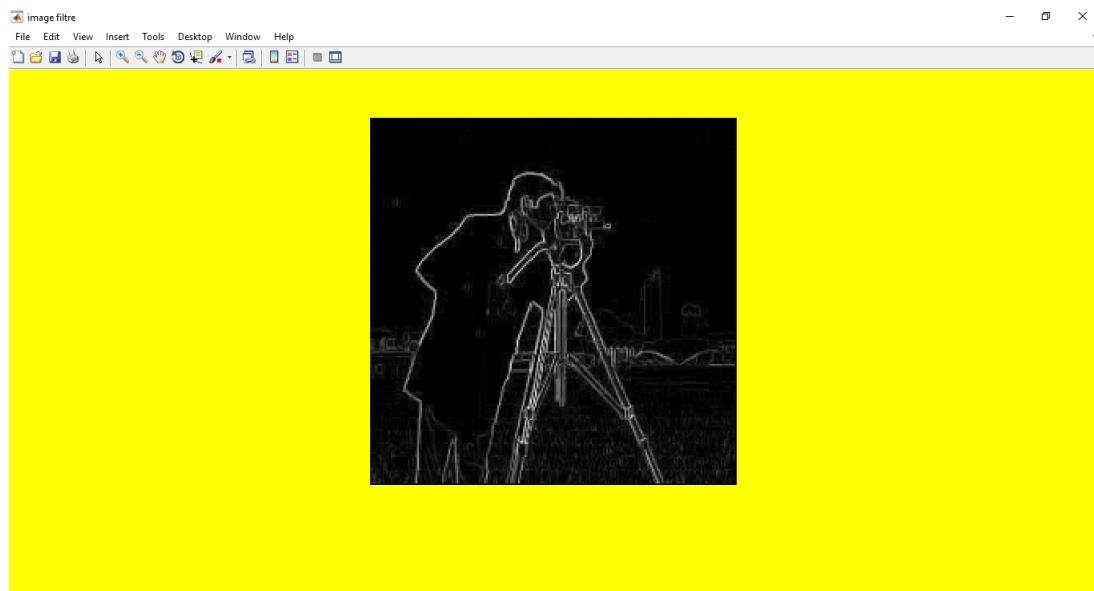


FIGURE 5.3 – figure agrandi

\* le groupe Options :

- **Type de contours** : le menu dans lequel l'utilisateur peut choisir l'opérateur de traitement.



FIGURE 5.4 – menu de types des contours

- **bruit** : menu qui permet de choisir le bruit à appliquer à l'image inserée.



FIGURE 5.5 – menu de types des bruit

- **seuil et seuillage** : Seuillage permet à l'utilisateur de faire son choix concernant le seuillage, s'il choisit oui il peut choisir la valeur de seuillage par le glisseur affiché au dessous :

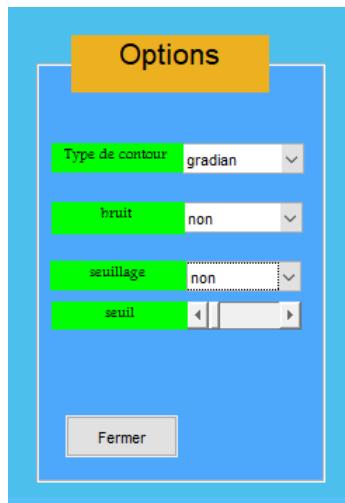


FIGURE 5.6 – le seuillage

- \* **Autres boutons réservés au filtre de sobel, au filtre de canny et au filtre de deriche :**

- *filtre de sobel* : si l'utilisateur choisit filtre de sobel un nouveau bouton s'affiche (Masque) pour choisir le type de masque à appliquer à l'image.



FIGURE 5.7 – paramètres de l’opérateur sobel

- *filtre de canny* : si l’utilisateur choisit filtre de canny trois nouveaux boutons s’affichent (sigma pour lissage gaussien, th pour le seuil haut et tb pour le seuil bas ).

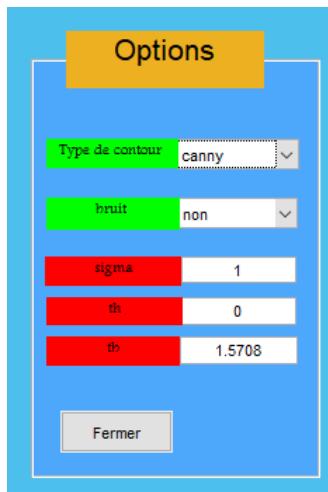


FIGURE 5.8 – paramètres de l’opérateur canny

- *filtre de deriche* : si l’utilisateur choisit l’opérateur de deriche un nouveau bouton s’affiche,c’est ”alpha” (facteur d’échelle).



FIGURE 5.9 – paramètres de l’opérateur deriche

## 5.5 Conclusion

Durant ce chapitre on a l’occasion de se rapprocher du monde de traitement d’image en concevant un outil graphique qui posséde un espace assez pratique qui pourra aider les utilisateurs à savoir les méthodes d’extraction des contours.

Cette application épurée, facile d’utilisation et intuitive .Son défaut, cependant, est de ne pas être portable.

## Conclusion générale

Au bout de notre cursus en licence informatique, nous avons été chargés de réaliser un projet de fin d'études. Notre travail s'est basé sur le traitement de l'image plus précisément la détection de contour. Au départ, nous n'avons aucune connaissances à ce domaine .Maintenant nous sommes très satisfaits d'avoir pu apprendre les bases de cette discipline et enrichir notre savoir et notre expérience.

Nous avons introduit dans ce rapport les notions et les opérations de base (convolution, filtrage, etc...) qui servent à la compréhension de différentes techniques de traitement d'images .Plusieurs méthodes classiques de ce dernier 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. .Puis nous avons montré quelques généralités sur le langage Matlab et un certain nombre de techniques de détection de contours sur les images à niveaux de gris. Ensuite nous avons mis en place une application qui permet de détecter les contours.

Ce mémoire nous a mené à montrer les performances de différents techniques de détection de contours. Toutefois ces méthodes ne donnent pas les mêmes résultats et certaines sont plus adaptées à un tel type d'image plutôt qu'un autre. D'où la nécessite de bien choisir la méthode plus adaptée à l'image considérée .Nous avons conclure qu'il n'existe pas de méthode universelle parfaite de détection de contour et que l'adéquation d'une méthode à un type d'images reste un problème largement ouvert.

## Bibliographie

- \* Cours "Traitement numérique d'images, MASTER SIDI (Systèmes d'Informations Décisionnels et Imagerie)", Pr. Aziz BAATAQUI et Pr. B. BOUDA, Université Moulay Ismail Faculté des Sciences et Technique, Errachidia.
- \* Bendaoud Mohammed Habib, thèse : développement de méthodes d'extraction de contours sur des images à niveau de gris.
- \* <http://www.wikipedia.com>
- \* <http://www.youtube.com>
- \* [www.developpez.com](http://www.developpez.com)
- \* [www.openclassrooms.com](http://www.openclassrooms.com)
- \* [www.mathworks.com](http://www.mathworks.com)