N° d'ordre : 07/DEA/TCO Année Universitaire : 2009/2010

**UNIVERSITE D'ANTANANARIVO**
---
**ECOLE SUPERIEURE POLYTECHNIQUE**
---
**DEPARTEMENT TELECOMMUNICATION**

MEMOIRE DE FIN D'ETUDES

en vue de l'obtention

du DIPLOME d'ETUDES APPROFONDIES

*Spécialité* : Télécommunication

*par* : **RAFIDISON Maminiaina Alphonse**

# *CONTRIBUTION A L'AMELIORATION DE LA SEGMENTATION D'IMAGES PAR CROISSANCE REGIONS*

Soutenu le 15 avril 2011 devant la Commission d'Examen composée de :

Président : M. RAKOTOMIRAHO Soloniaina

Examinateurs :

Mme RABEHERIMANANA Lyliane
M. RANDRIAMITANTSOA Paul Auguste
M. RAZAKARIVONY Jules

Directeurs de Mémoire :

M. RANDIMBINDRAINIBE Falimanana
M. RAZAFINDRADINA Henri Bruno

*« Si l'Eternel ne bâtit la maison. Ceux qui la bâtissent travaillent en vain ;*
*Si l'Eternel ne garde la ville. Celui qui la garde veille en vain. »*

***Psaume 127 :1***

## REMERCIEMENTS

Tout d’abord, j’aimerais remercier le Seigneur de m’avoir toujours donné la force de mener à bien la réalisation de ce travail de mémoire de fin d’études.

Je remercie, Le Directeur de l’Ecole Supérieure Polytechnique d’Antananarivo, Monsieur ANDRIANARY Philippe et Le chef de Département Télécommunication, Membre de jury, Monsieur RAZAKARIVONY Jules, pour ses accueils chaleureux au sein de l’établissement.

Je tiens à exprimer ma profonde et très sincère reconnaissance à Monsieur RAZAFINDRADINA Bruno, Maître de Conférences à l’IST d’Antsiranana et à Monsieur RANDIMBINDRAINIBE Falimanana, Maître de Conférences, Enseignant au sein du Département Télécommunication pour m’avoir encadré et qui n’ont cessé de me prodiguer de précieux conseils. Je les remercie pour la confiance qu’ils m’ont témoignée et pour m'avoir fait bénéficier de ses compétences et de ses conseils.

Je tiens aussi à remercier Monsieur RAKOTOMIRAHO Soloniaina, Professeur, Enseignant au sein du Département Télécommunication, qui nous a fait l’honneur de présider les membres du Jury de ce mémoire.

Je témoigne toute ma reconnaissance aux autres membres du jury qui ont voulu examiner ce travail :

- Mme RABEHERIMANANA Lyliane, Maître de Conférences, Enseignant au sein du Département Télécommunication.
- M. RANDRIAMITANTSOA Paul Auguste, Professeur, Enseignant au sein du Département Télécommunication.

J'adresse tout naturellement mes remerciements à tous les Enseignants de l’Ecole Supérieure Polytechnique en particulier les enseignants du troisième cycle en télécommunication, qui nous ont formés tout au long de l’année universitaire.

J'exprime ma très grande gratitude à ma famille, pour m'avoir soutenu durant mes études. Je les remercie d'avoir appuyé mes choix et d'avoir toujours su m'encourager.

Enfin, je ne saurai oublier toutes les personnes qui m’ont aidée de près ou de loin dans l’élaboration du présent mémoire.

# TABLE DES MATIERES

## NOTATIONS

1. Minuscules latines :

| | |
|---|---|
| $a_x, b_x$ | Niveaux de gris |
| $c_i$ | Contour numéro i |
| $d(x_j^{(i)}, y_i)$ | Distance entre $x_j^{(i)}$ et $y_i$. |
| $f$ | Fonction |
| $f'$ | Dérivée première de *f* |
| $g_{\max}$ | Niveau de gris max |
| $g_{\min}$ | Niveau de gris min |
| $h(x, y)$ | Noyau d'un filtre |
| $h(i)$ | Fréquence d'apparition du niveau de gris *i* |
| *max* | Valeur maximale |
| *min* | Valeur minimale |
| *n* | Ligne |
| *p* | Colonne |
| *r* | Région |
| $r^*$ | Région bien segmentée |
| *t* | Instant *t* |
| $t+1$ | Instant *t* + 1 |
| $v_i$ | Pixel au voisinage de *i* |
| $v(s,t)$ | Courbe ouverte ou fermée paramétrée par l'abscisse curviligne s et le temps t |
| *x* | Abscisse |
| $x_j^{(i)}$ | Pixel j appartenant à la classe i |
| *y* | Ordonnée |
| $y_i$ | Centroïde de la classe i |

2. Majuscules latines :

| | |
|---|---|
| $A$ | Arête |
| $A_x$ | Zone voisine |
| $B$ | Bleu |
| $C$ | Contraste/ Classe |
| $C^{'}$ | Nouvelle classe |
| $CR$ | Carte de région |
| $C^{[n]}$ | Couronne courante |
| $D$ | Distorsion globale |
| $D_{2H}$ | Dérivée seconde horizontale |
| $D_{2V}$ | Dérivée seconde verticale |
| $D_H$ | Dérivé horizontale |
| $D_V$ | Dérivée verticale |
| $D_i$ | Distance totale de la classe i, |
| $E_{ext}(c)$ | Energie externe |
| $E_{\text{int}}(c)$ | Energie interne |
| $E(c)$ | Energie totale |
| $G$ | Gradient |
| $I_e$ | Image d'entrée |
| $I_p$ | Image de polarité |
| $I_s$ | Image de sortie |
| $I_z$ | Image de passage à zéro |
| $I(x, y)$ | Intensité de niveau de gris au point (x,y) |
| $Lum$ | Luminance |
| $L$ | Laplacien |
| $L_i$ | Nombre de points de la classe i, |
| $L_x$ | Degré de luminosité / Laplacien |
| $M$ | Nombre de colonnes, |

| | |
|---|---|
| $M_x$ | Masque direction verticale |
| $M_y$ | Masque direction horizontale |
| $N$ | Nombre de lignes, |
| $N(I)$ | Négatif de l'image I |
| $N(x,y)/G(x,y)$ | Norme du gradient au point (x,y) |
| $R$ | Rouge |
| $R^{[0]}$ | Région initiale |
| $R^{[n]}$ | Région courante |
| $R^{[n+1]}$ | Région suivante |
| $S(i, j)$ | Seuil à appliquer au point (i,j) |
| $S_H$ | Dérivée horizontale de Sobel |
| $S_O$ | Dérivée oblique de Sobel |
| $S_V$ | Dérivée verticale de Sobel |
| $T^*$ | Opérateur dual |
| $U_s$ | Seuil pour séparation des objets du fond |
| V | Vert |

3. Minuscules grecques :

| | |
|---|---|
| $\alpha , \beta$ | Translation |
| $\alpha_{mal\_classés}$ | Pourcentage de pixels mal classés |
| $\alpha_{bien\_classés}$ | Pourcentage de pixels bien classés |
| $\theta$ | Orientation de contour |
| $\mu$ | Valeur moyenne |
| $\sigma$ | Ecart type |
| $\sigma w^2$ | Variance intra-classes |

4. Majuscules grecques :

| | |
|---|---|
| $\nabla f(x, y)$ | Gradient de $f(x, y)$ |

5. Abréviations :

| | |
|---|---|
| Card | Cardinalité |
| CCD | Charge Coupled Device |
| DPI | Dots Per Inch |
| FN | False Negative |
| FP | False Positive |
| KI | Kappa Index |
| NPBC | Nombre de Pixel Bien Classé |
| NPMC | Nombre de Pixel Mal Classé |
| NTP | Nombre Total de Pixel |
| Pixel | Picture Element |
| PPP | Points Par Pouce |
| RVB | Rouge Vert Bleu |
| TF | Transformé de Fourier |
| TP | True Positive |
| TV | Télévision |

6. Notations spéciales :

| | |
|---|---|
| * | Convolution |
| $\frac{\partial f}{\partial x}$ | Dérivée de f par rapport à x |
| $\frac{\partial^2 f}{\partial^2 x}$ | Dérivée seconde de f par rapport à x |

# INTRODUCTION GENERALE

Le traitement d'images est né dans les années 1920 pour la transmission d'images par le câble sous-marin de New York à Londres. Harry G. Bartholomew et Maynard D. McFarlane ont effectué la première numérisation d'image avec compression de données pour envoyer des fax de Londres à New York. Le temps de transfert passait ainsi de plus d'une semaine à moins de trois heures. Il n'y avait pas vraiment d'évolution par la suite jusqu'à la période d'après-guerre.

Le véritable essor du traitement d'images n'a eu lieu que dans les années 1960 quand les ordinateurs commençaient à être suffisamment puissants pour travailler sur des images. L'enseignant, Marvin Minsky avait donné à son élève Gerald Sussman, un travail d'été qui consistait à faire « voir » un ordinateur. Sussman fut écœuré et Minsky qui a reçu le plus gros budget de l'époque n'a pas réussi à synthétiser son observateur-descripteur. A partir de cette époque là, les scientifiques commencent à apprécier leur ordinateur, à tel point qu'ils y voyaient en toute confiance la possibilité de simuler l'intelligence.

La vision par ordinateur est depuis devenue un domaine de recherche indépendant, et des plus prolifiques. David Maar a distingué trois niveaux de traitement de l'information visuelle: le premier traitement étant l'extraction de caractéristiques; le deuxième c'est la mise en relation de ces caractéristiques entre elles en incluant le point de vue de l'observateur; et enfin la description de la scène en terme d'objets et de relations entre objets, indépendamment du point de vue.

La segmentation représente l'ambition de la première étape qui n'est autre que détecter les objets présents dans une image. La diversité des images et la difficulté du problème ont conduit à l'introduction d'une multitude d'algorithmes que nous n'aurons pas ici l'ambition de résumer. Les résultats obtenus, évalués souvent de façon très empirique, ainsi que l'évolution des capacités de traitement des ordinateurs, ont souvent amené des scientifiques de profils variés à traiter le problème de manières très différentes.

L'objectif de ce présent mémoire est donc de trouver une méthode un peu plus efficace par rapport aux algorithmes de segmentation d'images existants. Le démarche de présentation de notre ouvrage se fait en trois grands chapitres et se présente comme suit:

- Le premier chapitre : Notions fondamentales sur le traitement d'images ; ce chapitre décrit les terminologies utiles dans le domaine de traitement d'images et les différentes techniques de traitements.
- Le deuxième chapitre : Etat de l'art de segmentation d'images ; nous décrivons les méthodes de segmentation existantes afin qu'on puisse faire des comparaisons dans le but d'offrir une nouvelle technique plus performante.
- Le troisième chapitre : Nouveau concept de segmentation d'images.

# CHAPITRE 1
# NOTIONS FONDAMENTALES SUR LE TRAITEMENT D'IMAGES

## 1.1. Introduction

L'être humain, de par sa nature, éprouvait le besoin d'échanger de l'information depuis la nuit des temps. Il inventa des méthodes aussi diverses les unes des autres pour pouvoir communiquer. La perception visuelle était son outil le plus exploité, et le moyen de communication le plus utilisé était le dessin [15].

Donc, l'image était, depuis l'aube de l'humanité, un élément des plus significatifs dans le domaine de l'échange de l'information, et elle l'est toujours d'ailleurs. Sa puissance réside dans le fait qu'une image, à elle seule, puisse remplacer des quantités considérables de mots, et ce, en allant des images monochromes (noir et blanc) qui sont les plus simples mais aussi les moins porteuses d'informations, aux images en couleurs vraies, exploitées actuellement par de puissants calculateurs et servant dans des domaines de pointe ; à savoir : la sécurité, la médecine, l'astronomie, le multimédia et bien d'autres.

Ainsi, depuis les débuts du traitement d'images jusqu'à nos jours, de nombreux chercheurs se sont penchés sur la question de l'imagerie, afin de permettre une meilleure exploitation de cette forme de message, sans pour autant subir les contraintes liées à l'énormité de l'espace de stockage requis.

Nous allons exposer certains concepts élémentaires s'accordant au domaine de l'imagerie et du traitement d'images.

## 1.2. Définition de l'image

L'image est une représentation d'une personne ou d'un objet par la peinture, la sculpture, le dessin, la photographie, etc. [07]

C'est aussi un ensemble structuré d'informations qui, après affichage sur l'écran, ont une signification pour l'œil humain.

Elle peut être décrite sous la forme d'une fonction I(x,y) de brillance analogique continue, définie dans un domaine borné, tel que x et y sont les coordonnées spatiales d'un point de l'image et I est une fonction d'intensité lumineuse et de couleur. Sous cet aspect, l'image est inexploitable par la machine, ce qui nécessite sa numérisation [02].

### 1.3. Image numérique

Contrairement aux images obtenues à l'aide d'un appareil photo analogique, ou dessinées sur du papier, les images manipulées par un ordinateur sont numériques (représentées par une série de bits). 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. La numérisation d'une image est la conversion de celle-ci de son état analogique (distribution continue d'intensités lumineuses) en une image numérique représentée par une matrice bidimensionnelle de valeurs numériques I(x,y) où x, y : coordonnées cartésiennes d'un point de l'image et I(x, y) : niveau de gris en ce point.

Pour des raisons de commodité de représentation pour l'affichage et l'adressage, les données images sont généralement rangées sous formes de tableau I de n lignes et p colonnes. Chaque élément I(x ,y) représente un pixel de l'image et à sa valeur est associé un niveau de gris codé sur m bits ($2^m$ niveaux de gris ; 0 = noir ; $2^m$-1 = blanc). La valeur en chaque point exprime la mesure d'intensité lumineuse perçue par le capteur [02].

### 1.4. Caractéristique d'une image

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

#### *1.4.1. Pixel*

Une image est constituée d'un ensemble de points appelés pixels. Le mot pixel provient d'une contraction et de la juxtaposition de l'expression britannique "PICture ELement". Le pixel représente ainsi le plus petit élément constitutif d'une image numérique auquel on peut associer individuellement une couleur (ou un niveau de gris) et une intensité [02].

L'ensemble de ces pixels est contenu dans un tableau à deux dimensions constituant l'image finalement obtenue.

Chaque pixel est défini dans une image par des coordonnées et étant donné que l'écran effectue un balayage de gauche à droite et de haut en bas, on désigne généralement par les coordonnées (0,0) le pixel situé en haut à gauche de l'image.

#### *1.4.2. Définition et résolution*

On appelle *définition* le nombre de points (pixel) constituant l'image, c'est-à-dire sa «dimension informatique» (le nombre de colonnes de l'image que multiplie son nombre de lignes). Une image

possédant 640 pixels en largeur et 480 en hauteur aura une définition de 640 pixels par 480, notée 640x480.

La *résolution*, détermine par contre le nombre de points par unité de surface, exprimé en points par pouce (PPP, en anglais DPI pour Dots Per Inch); un pouce représentant 2,54 cm.

La résolution permet ainsi d'établir le rapport entre le nombre de pixels d'une image et la taille réelle de sa représentation sur un support physique. Une résolution de 300 dpi signifie donc 300 colonnes et 300 rangées de pixels sur un pouce carré ce qui donne donc 90000 pixels sur un pouce carré.

### *1.4.3. Bruit*

Le bruit est tout phénomène imprévisible qui vient perturber le signal. Dans une image c'est un phénomène de brusques variations d'intensité d'un pixel par rapport à ses voisins.

Le bruit peut être causé par :

- Les événements inattendus lors de l'acquisition comme le bougé ou une modification ponctuelle des conditions d'éclairage;
- La mauvaise qualité des capteurs ou une mauvaise utilisation de ces derniers ;
- Lors de l'échantillonnage. Le passage de la forme analogique à la forme numérique de l'image ;
- Ou bien la nature de la scène elle même (poussières, rayures, perturbation atmosphérique,…).

### *1.4.4. Niveau de gris*

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.

Le nombre de niveaux de gris dépend du nombre de bits utilisés pour décrire la " couleur " de chaque pixel de l'image. Plus ce nombre est important, plus les niveaux possibles sont nombreux.

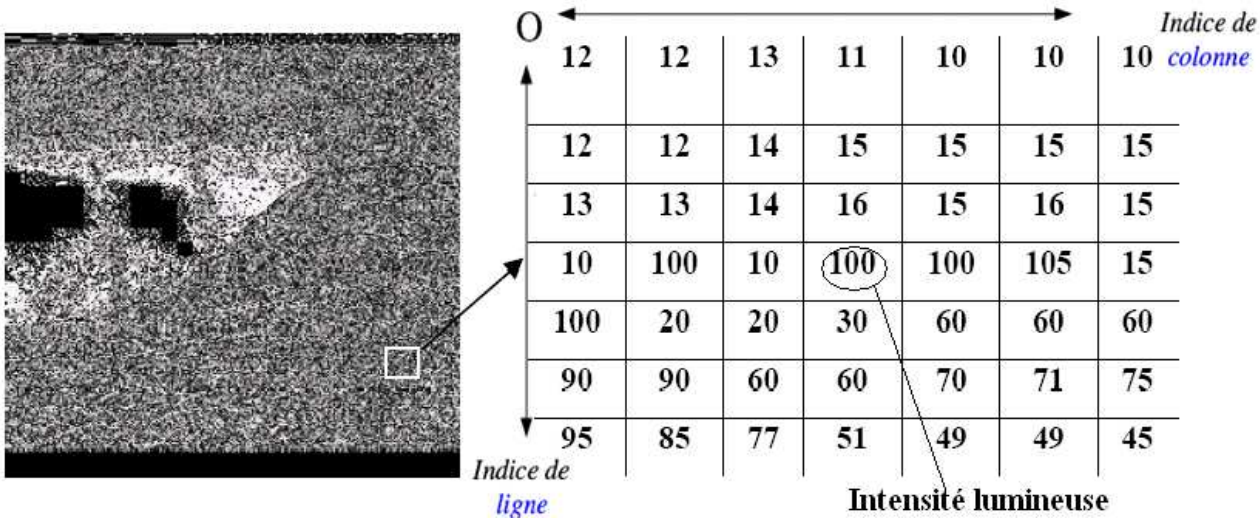


*Figure 1.01* : exemple d'image en niveau de gris et sa quantification

### *1.4.5. Couleur*

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. Les systèmes émettant de la lumière (écrans d'ordinateurs,...) sont basés sur le principe de la synthèse additive : les couleurs sont composées d'un mélange de rouge, vert et bleu (modèle R.V.B.).

#### 1.4.5.1. Représentation en couleurs réelle

Elle consiste à utiliser 24 bits pour chaque point de l'image. 8 bits sont employés pour décrire la composante rouge (R), 8 pour le vert (V) et 8 pour le bleu (B). Il est ainsi possible de représenter environ $256^3$=16,7 millions de couleurs différentes simultanément. Cela est cependant théorique, car aucun écran n'est capable d'afficher 16 millions de points. Dans la plus haute résolution actuelle (1600 x 1200), l'écran n'affiche que 1 920 000 points. Par ailleurs, l'œil humain n'est pas capable de distinguer autant de couleurs.

#### 1.4.5.2. Représentation en couleur indexée

Afin de diminuer la charge de travail nécessaire pour manipuler des images en 24 bits, on peut utiliser le mode de représentation en couleurs indexée. Le principe consiste à déterminer le

nombre de couleurs différentes utilisées dans l'image, puis à créer une table de ces couleurs en attribuant à chacune une valeur numérique correspondant à sa position dans la table. La table, appelée *palette*, comporte également la description de chacune des couleurs, sur 24 bits.

### *1.4.6. 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. Pour diminuer l'erreur de quantification, pour comparer deux images obtenues sous des éclairages différents, ou encore pour mesurer certaines propriétés sur une image, on modifie souvent l'histogramme correspondant.

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 les 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.

### *1.4.7. Contraste*

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 :

$$C = \frac{L1 - L2}{L1 + L2} \qquad (1.01)$$

### *1.4.8. Luminance*

C'est le degré de luminosité des points de l'image. Elle est définie aussi comme étant le quotient de l'intensité lumineuse d'une surface par l'aire apparente de cette surface, pour un observateur lointain, le mot luminance est substitué au mot brillance, qui correspond à l'éclat d'un objet.

La moyenne ou luminance (brillance) d'une image numérique en niveau de gris est définie comme la moyenne des pixels de l'image :

$$Lum(I) = \frac{1}{MxN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} I(x, y) \qquad (1.02)$$

Avec :

M : Nombre de colonnes,

N : Nombre de lignes,

$I(x, y)$ : La valeur de niveau de gris au point $(x, y)$

### *1.4.9. Région*

C'est un ensemble de pixels connexes et homogènes. Un pixel appartient à une région donnée s'il vérifie les caractéristiques de celle-ci (intensité, …). Une région est limitée par un contour.

### *1.4.10. Contour et texture*

Les contours représentent la frontière entre les objets de l'image, ou la limite entre deux pixels dont les niveaux de gris représentent une différence significative. Les textures décrivent la structure de ceux-ci.

L'extraction de contour consiste à identifier dans l'image les points qui séparent deux textures différentes.

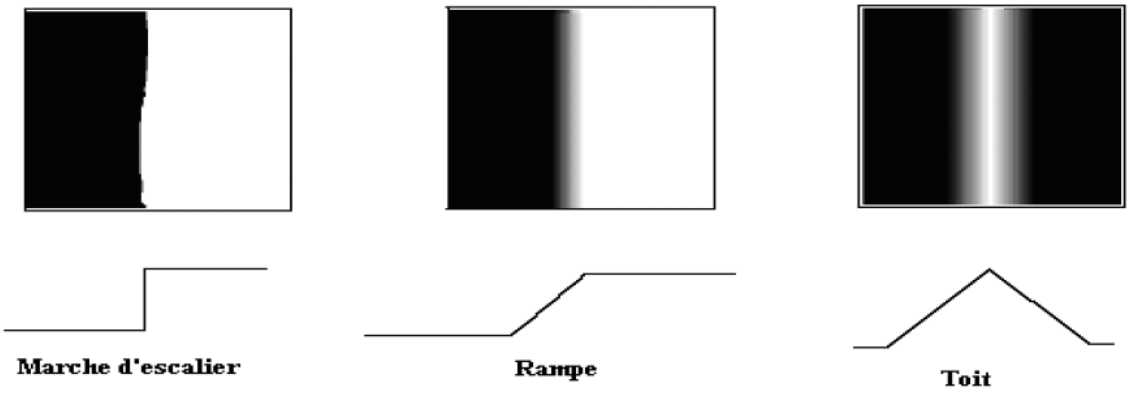


*Figure 1.02* : différents type de contours

### 1.5. Système de traitement d'image

Avant d'être visualisé, une image passera dans quatre grandes étapes comme montré sur la figure ci-dessous.

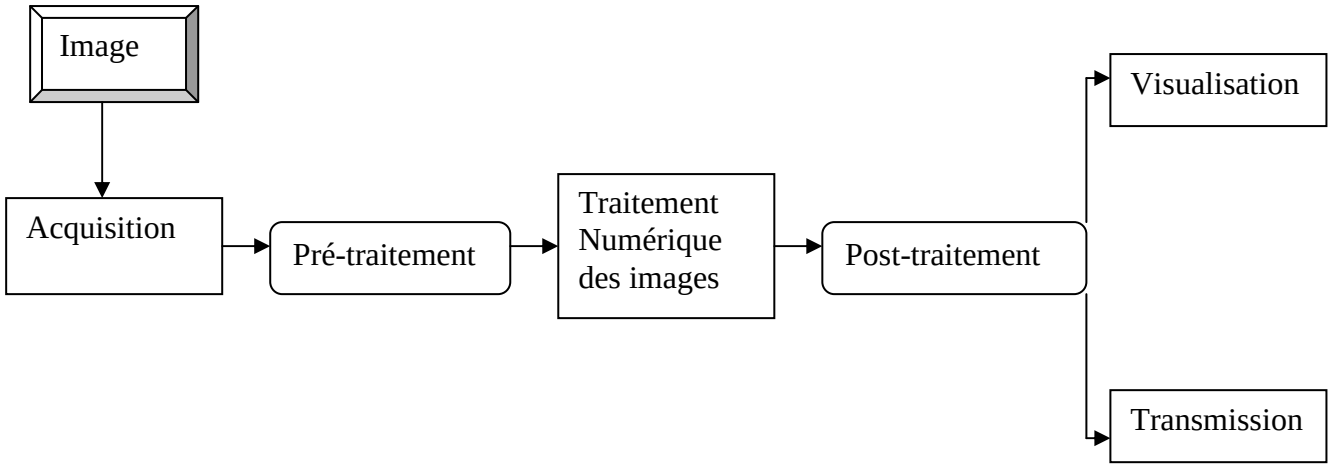


*Figure 1.03* : système d'acquisition d'image

#### *1.5.1. Acquisition des données images*

L'acquisition d'images constitue un des maillons essentiels de toute chaîne de conception et de production d'images. Pour pouvoir manipuler une image sur un système informatique, il est avant tout nécessaire de lui faire subir une transformation qui la rendra lisible et manipulable par ce système. Le passage de cet objet externe (l'image d'origine) à sa représentation interne (dans l'unité de traitement) se fait grâce à une procédure de numérisation. Ces systèmes de saisie, dénommés optiques, peuvent être classés en deux catégories principales : les caméras numériques et les scanners [02].

A ce niveau, notons que le principe utilisé par le scanner est de plus en plus adapté aux domaines professionnels utilisant le traitement de l'image comme la télédétection, les arts graphiques, la médecine, etc.

Le développement technologique a permis l'apparition de nouveaux périphériques d'acquisition appelés cartes d'acquisition, qui fonctionnent à l'instar des caméras vidéo, grâce à un capteur CCD (Charge Coupled Device). La carte d'acquisition reçoit les images de la camera, de la TV ou du scanner afin de les convertir en informations binaires qui seront stockées dans un fichier.

### *1.5.2. Dispositifs de numérisation d'images*

Suivant l'objet ou le document à numériser et le domaine d'application dans lequel l'image va être utilisée, il existe divers dispositifs de numérisation d'images allant du simple scanner à main au satellite de télédétection.

### *1.5.3. Pré-traitement et post-traitement*

Le pré-traitement concerne les images acquises tandis que le post-traitement concerne les images traitées. Dans les deux cas, les images ne peuvent échapper aux effets de dégradations dus essentiellement aux phénomènes physiques tels que :

- La diffraction (déviation) du système optique.
- Le flou dû au mouvement de l'image durant son acquisition.

Pour pallier à ces dégradations on utilise en général le filtrage.

## 1.6. Domaine d'application

Le traitement d'images possède l'aspect multidisciplinaire. On trouve ses applications dans des domaines très variés tels que: Médecine (Radiographies, …), Biologie, Météorologie, Astronomie, Géologie, Physique (spectroscopie, physique des plasmas,…), Applications militaires, Applications industrielles (Robotique, surveillance de qualité,…), Photographies, Publicité, etc. [02][14][15]

## 1.7. Numérisation des images

Le processus de numérisation se décompose en 3 étapes comme décrit dans la figure ci-dessous :

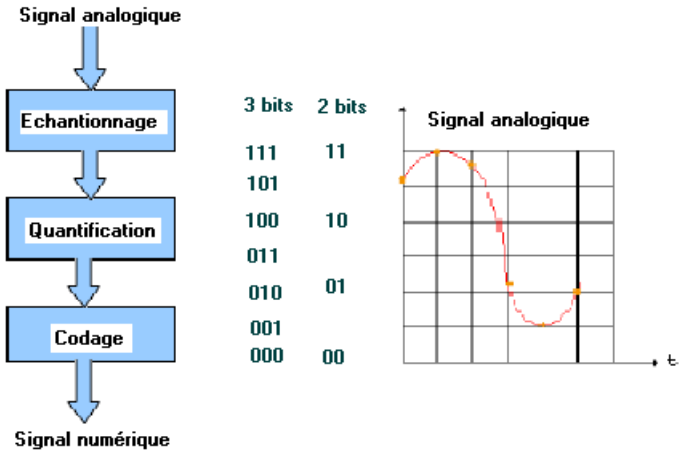


*Figure 1.04* : système de numérisation

### *1.7.1. Balayage*

La première étape consiste à balayer l'image, ligne par ligne, tout en captant des mesures (luminosité, intensité). Ce balayage nous renvoie le spectre de l'image qui est une fonction continue (fonction de l'image).L'aspect analogique est, donc, toujours présent [14].

Cette fonction est traitée, par la suite, de façon à nous fournir des valeurs numériques représentatives de l'image.

### *1.7.2. Echantillonnage*

C'est la deuxième étape du processus, elle consiste à prendre des valeurs numériques discrètes pour chaque intervalle défini par la taille de l'ouverture de balayage. Ainsi, à chaque intervalle, on garde la valeur de la fonction spectrale résultante du balayage. Pour faciliter la compréhension de cette notion, on dira que cet échantillonnage représente le découpage de l'image d'origine en petits carrés où chaque carré est caractérisé par la mesure évaluée dans l'étape du balayage. Ainsi, l'image sera transformée en une mosaïque de ces petits carrés.

### *1.7.3. Quantification et codage*

Il s'agit, durant cette étape, de codifier les valeurs des carrés, créées à partir de la seconde étape, en nombres entiers qui seront à leurs tours codifiés sous forme binaire.

## 1.8. Méthodes de traitement d'images

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 [02].

### *1.8.1. Opérateur statistique*

La description d'une forme nécessite un certain nombre d'opération qui transforme l'image originale en une représentation compatible au traitement ultérieur. Cette représentation donne les caractéristiques d'une image exprimée sous forme numérique ou structurelle.

1.8.1.1. Quelques transformations des niveaux de gris d'une image

On considère les images à niveaux de gris compris entre 0 et $M$ (par exemple, $M = 255$). Pour traiter le contraste d'une image, on applique uniformément à tous les pixels de celle-ci une

transformation de niveaux de gris *f*. En d'autres termes, à partir d'une image *I* associant à tout pixel *p* le niveau de gris *I*(*p*), on obtient une nouvelle image *J* dont le niveau de gris au pixel *p* est *J*(*p*) = *f*(*I*(*p*)).

*1.8.1.1.1. Etirement de contraste*

Supposons que dans une image donnée, l'intervalle des niveaux de gris utilisés ne s'étende pas de 0 à *M*. Soient *gmin* et *gmax* les niveaux de gris minimum et maximum des pixels de l'image. L'*étirement de contraste* consiste en l'application aux niveaux de gris de l'image d'une fonction linéaire *f* telle que *f*(*gmin*) = 0 et *f*(*gmax*) = *M*. On a ainsi :

$$f(x) = M\,(x - gmin) / (gmax - gmin) \qquad (1.03)$$

*1.8.1.1.2. Négatif*

Le *négatif N*(*I*) d'une image *I* est obtenu en inversant l'échelle de niveaux de gris de 0 à *M* ; en d'autres termes, en tout pixel *p* on a :

$$N(I(p)) = M - I(p) \qquad (1.04)$$

Un usage pratique du négatif est la production de diapositives par photographie d'écran. Un film pour diapositives est généralement plus cher et de moindre qualité qu'un film traditionnel. En affichant une image en négatif et en la photographiant avec un film classique, le développement du négatif donnera un négatif de négatif de l'image, c'est-à-dire un positif, qui pourra être utilisé comme diapositive. Cela marche très bien pour des images à niveaux de gris en utilisant un film noir sur blanc (c'est-à-dire à niveaux de gris).

Du point de vue théorique, le négatif est utilisé pour définir le *dual* (par inversion de niveaux de gris) d'une opération de traitement d'images. Soit *T* une opération de traitement d'images, transformant une image *I* en une image *T*(*I*) ; le *dual T** de *T* est défini comme l'application d'un négatif, suivi de *T*, puis d'un deuxième négatif :

$$T^*(I) = N(T(N(I))) \qquad (1.05)$$

On a donc *N*(*T**(*I*)) =*T*(*N*(*I*)), en d'autres termes *T** est l'opération correspondant à l'application de *T* sur le négatif de l'image. Donc *T** se comportera sur les zones claires de l'image comme *T* sur les zones sombres, et vice versa.

*1.8.1.1.3. Marquage de zone*

Cette opération remplace un intervalle de niveaux de gris par du blanc (niveaux de gris *M*) ou du noir (niveaux de gris 0). La transformation de niveaux de gris correspondante est illustrée ci-dessous :

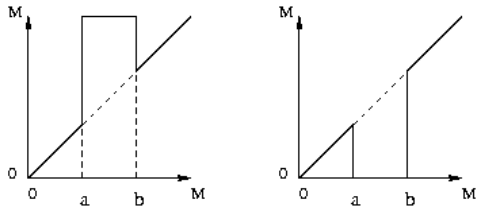


*Figure 1.05* : masquages blanc et noir de l'intervalle [a,b]

Cette opération permet de mettre en évidence ou au contraire de masquer certaines parties de l'image dont les niveaux de gris se trouvent dans une plage donnée.

*1.8.1.1.4. Rehaussement des niveaux de gris*

Soit *M* le niveau de gris maximum (par exemple, *M* = 255). On suppose que le niveau de gris minimum est 0. Le rehaussement consiste à appliquer aux niveaux de gris de l'image une fonction croissante *f* telle que *f*(0) = 0 et *f*(*M*) = *M*.

Le graphe de *f* indique le type de rehaussement appliqué à l'image: les plages de niveaux de gris où la pente de *f* est forte sera rehaussée aux dépens de celles où cette pente est faible.

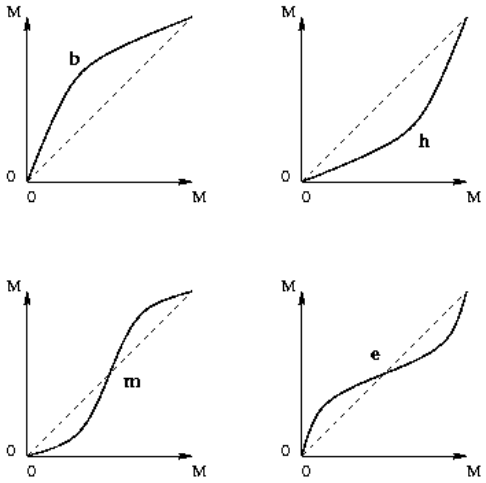


*Figure 1.06* : exemple de fonction de rehaussement

Par exemple dans l'illustration ci-dessus, *b* rehausse les bas niveaux de gris (zones sombres), tandis que *h* rehausse les hauts niveaux de gris (zones claires), *m* rehausse les niveaux de gris moyens, et *e* rehausse les niveaux de gris extrêmes.

*1.8.1.1.5. Rehaussement linéaire par morceau*

Dans bien des cas, on peut mesurer les niveaux de gris sur différentes zones de l'image correspondant à certains objets spécifiques, qu'on comparera à des valeurs souhaitables. Par exemple sur une photo d'identité à 256 niveaux de gris, on peut avoir les niveaux de gris suivants :

| ***Zone*** | ***niveau de gris mesuré*** | ***niveau de gris souhaitable*** |
|---|---|---|
| Fond | 105 | 70 |
| Cheveux | 120 | 105 |
| Chemise | 150 | 190 |
| Visage | 180 | 225 |

*Tableau 1.01 :* rehaussement linéaire par morceau

Pour rehausser l'image, on appliquera aux niveaux de gris une transformation f linéaire par morceaux définie par :

$f(0) = 0$,
$f(105) = 70$,
$f(120) = 105$,
$f(150) = 190$,
$f(180) = 225$,
$f(255) = 255$,

Avec une interpolation linéaire entre les points de contrôle 0, 105, 120, 150, 180, et 255, comme illustré sur l'image ci-dessous.

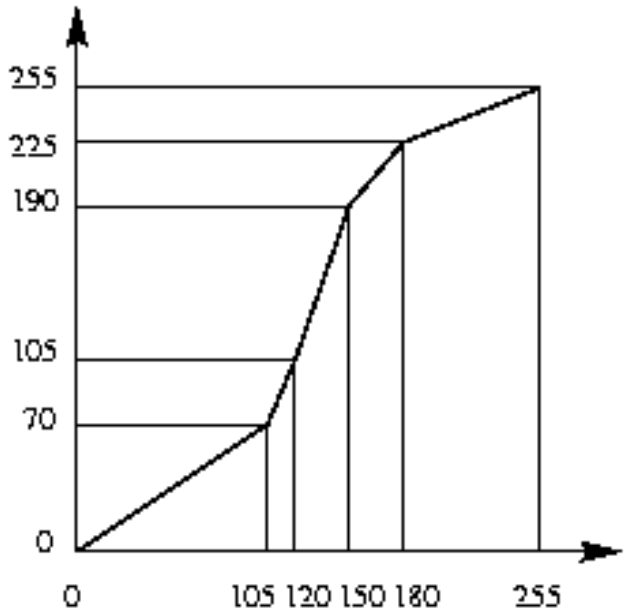


*Figure 1.07* : fonction de rehaussement linéaire par morceau

Plus généralement, pour une image à niveaux de gris compris entre 0 et $M$, on peut avoir $n$ niveaux de gris $a_1, ..., a_n$ (tels que $0 < a_1 < ... < a_n < M$), à transformer en $n$ niveaux de gris $b_1, ..., b_n$ (où l'on aura le plus souvent $0 < b_1 < ... < b_n < M$), et on choisira une transformation $f$ linéaire par morceaux donnée par :

$$f(0) = 0, \quad f(a_1) = b_1, \quad ..., \quad f(a_n) = b_n, \quad \text{et} \quad f(M) = M \qquad (1.06)$$

*1.8.1.1.6. Egalisation d'histogramme*

L'*égalisation* d'histogramme a pour but d'harmoniser la répartition des niveaux de luminosité de l'image, de telle manière à tendre vers un même nombre de pixel pour chacun des niveaux de l'histogramme. Cette opération vise à augmenter les nuances dans l'image. On égalise quand l'histogramme est étalé mais pas homogène, on obtient à la fin un histogramme à peu prés "plat".

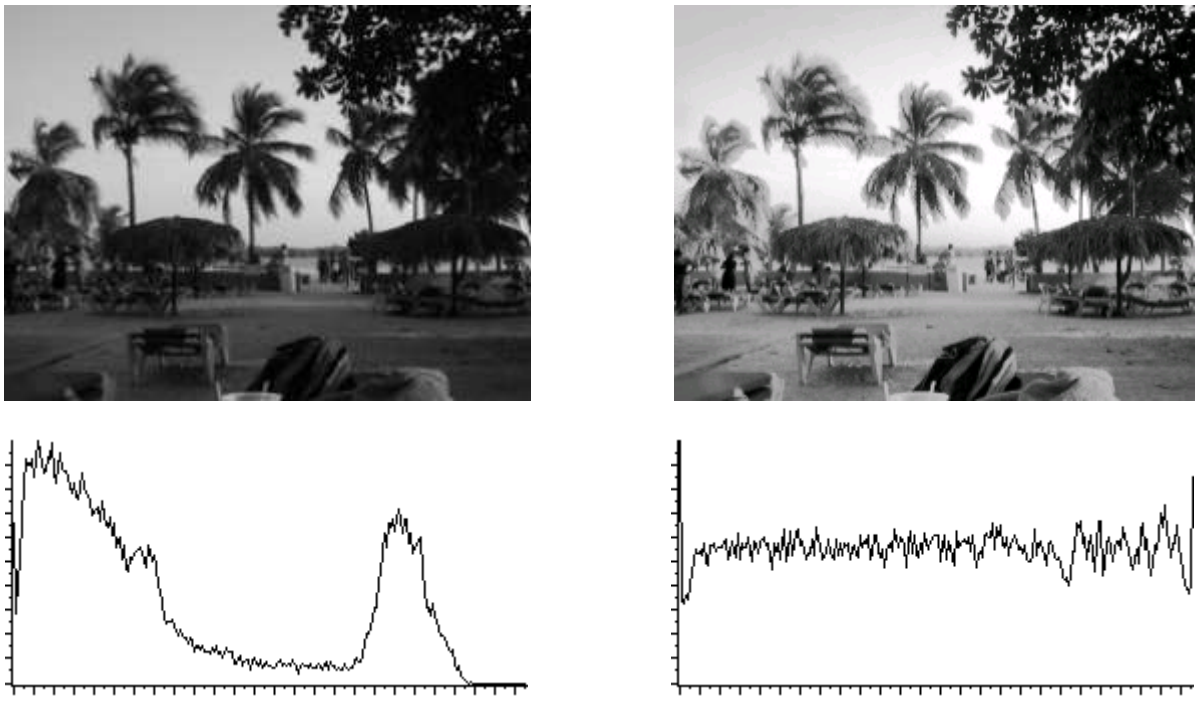

*Figure 1.08* : égalisation de l'histogramme

1.8.1.2. Séparation des objets du fond

Elle consiste à subdiviser une scène réelle en ses parties constituantes objet, en les isolant du fond dans le cas d'une image à forte contraste. Une valeur seuil $U_s$ sépare les deux groupes de niveaux de gris.

L'image seuillée ou binarisée est notée :

$$I(x,y)= \begin{cases} 1 & \text{si le niveau de gris } \geq U_s \quad \text{(objet)} \\ 0 & \text{sinon (fond)} \end{cases} \qquad (1.07)$$

$U_s$ est déterminé comme le premier minimum avant le dernier maximum.

### *1.8.2. Filtrage d'image*

L'amélioration d'une image est essentiellement obtenue par ce que l'on appelle une opération de filtrage. Le filtrage est utilisé pour modifier ou pour mettre en valeur une image par exemple. On pourra filtre une image pour accentuer certains attribues ou pour en supprimer d'autres. L'objectif du filtrage est d'éliminer les perturbations induites par les procédés d'acquisition d'image et au problème de transmission ainsi de réduire les variations d'intensité au sein de chaque région de

l'image tout en respectant l'intégrité de la scène originale comme les éléments significatifs et les transitions entre les régions homogènes.
Différentes méthodes de filtrage ont été développées suivant le type et l'intensité du bruit ou les applications auxquelles on destine l'image. La première est basée sur le filtrage linéaire, mais les limites atteintes par ces techniques ont conduit au développement des filtres non linéaires.

1.8.2.1. Filtrage linéaire

Le filtrage linéaire d'une image peut s'envisager de deux manières. Le filtrage peut tout d'abord se réaliser dans le domaine spatiale en effectuant un produit de convolution ou dans le domaine fréquentielle en multipliant la transformé de Fourier de l'image par la fonction de transfert du filtre. L'image filtrée est alors obtenue par la transformé inverse

*1.8.2.1.1. Filtrage par produit de convolution*

Comme dans le cas d'un système unidimensionnel, on peut interpréter l'action d'un opérateur bidimensionnel linéaire et invariant en translation comme un filtrage.
Si $I_{s1}(x,y)$ et $I_{s2}(x,y)$ sont les images de sortie du système relative aux images d'entrée $I_{e1}(x,y)$ et $I_{e2}(x,y)$, un système linéaire invariant dans le temps vérifie les propriétés suivantes :

- Linéarité : $aI_{e1}(x,y)+bI_{e2}(x,y) \rightarrow aI_{s1}(x,y)+bI_{s2}(x,y)$ (1.08)
- Invariance : $I_e(x-t,y-t) \rightarrow I_s(x-t,y-t)$ (1.09)

Un filtre linéaire est un exemple important du système linéaire. Il est utilisé pour modifier le contenu spectrale d'une image afin d'en retirer l'information recherchée.
Un filtre linéaire bidimensionnel est caractérisé par sa réponse impulsionnelle $h(x,y)$. L'image de sortie $I_s(x,y)$ résulte du produit de convolution de la réponse impulsionnelle du filtre par l'image d'entrée $I_e(x,y)$

$$I_e(x,y) \rightarrow I_s(x,y) = I_e(x,y) * h(x,y) \quad (1.10)$$

Contrairement à la manipulation d'histogramme qui est des opérations ponctuelles sur tous les pixels de l'image, le filtrage est un produit de convolution qui met en jeu l'environnement de chaque pixel voisin.

La convolution est définie par les relations suivantes :

A une dimension : $I_s(t) = I_e(t) * h(t) = \int_{-\infty}^{+\infty} I_e(\tau) h(t-\tau) d\tau$ (1.11)

A deux dimensions : $I_s(x,y) = I_e(x,y) * h(x,y) = \int_{-\infty-\infty}^{+\infty+\infty} I_e(\alpha,\beta) h(x-\alpha, y-\beta) d\alpha d\beta$ (1.12)

En ce qui concerne la convolution dans le domaine de traitement d'image, chaque pixel est remplacé par une combinaison linéaire des pixels qui l'entourent. Un produit de convolution est un opérateur mathématique que l'on utilise pour multiplier des matrices entre elle. Dans le cas qui nous intéresse, nous mettons en jeu deux matrices très différentes : la matrice image très grande et une matrice plus petite qu'on appelle le noyau parce que c'est le cœur de tout le changement qui affectera l'image et ce noyau représente le filtre.

Un filtre est défini par une fenêtre carrée de dimension impaire qui est déplacé sur l'image.

La convolution est représentée par la relation suivante :

$$I_s(x,y) = \sum_u \sum_v h(u,v) I_e(x-u, y-v) \quad (1.13)$$

$$= \sum_u \sum_v I_e(u,v) h(x-u, y-u) \quad (1.14)$$

$h(x,y)$ : Noyau du filtre

$I_e(x,y)$ : Image à filtrer

Pour mieux comprendre, illustrons la notion de filtrage à l'aide d'un petit exemple :

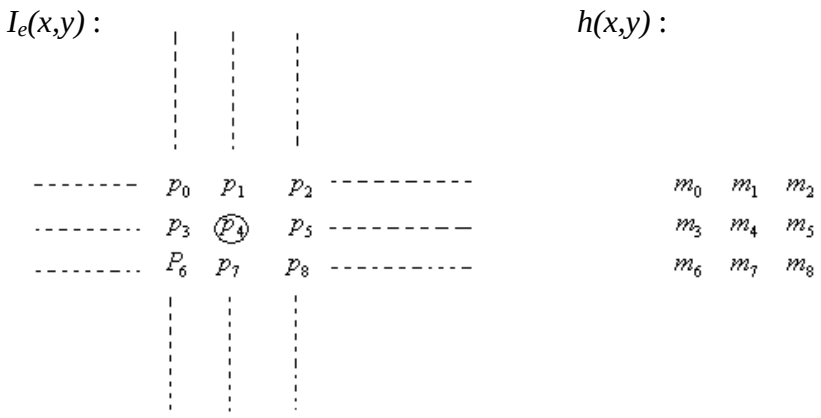


*Figure 1.09* : principe de filtrage

En appliquant la convolution, $p_4$ devient $p'_4 = m_0 p_0 + m_1 p_1 + ... + m_8 p_8$ (1.15)

Pour la convolution des bords, il y a l'ajout de zéro, l'enroulement, réflexion et par défaut.

*Quelques exemples de filtres*

- Lissage

C'est une opération destinée à éliminer les bruits dans une image. Les lissages sont des filtres passe-bas ce qui signifie qu'ils éliminent les signaux de haute fréquence caractérisés par des grandes variations de niveau de gris entre pixel voisin.

Pour un lissage fort :

$$\text{h(x,y)} : \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad (1.16)$$

Pour un lissage moyen :

$$\text{h(x,y)} : \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \quad (1.17)$$

- Opérateur de dérivation

La dérivée première : en développant au premier ordre, avec une approximation, la dérivée en un point $f'(x)$ peut s'exprimer par :

$f'(x) = \frac{1}{2}[f(x+1) - f(x-1)]$ (1.18) appliqué à une image numérique, on peut définir une dérivée partielle $\frac{df(x,y)}{dx}$ suivant les colonnes par le filtre de matrice suivante :

$$D_H = \frac{1}{2}\begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \text{ Dérivée horizontale} \quad (1.19)$$

La dérivée verticale dérivée partielle $\frac{df(x,y)}{dy}$ suivant les lignes d'image est défini par la matrice :

$$D_V = \frac{1}{2}\begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \text{ Dérivation verticale} \quad (1.20)$$

Opérateur de Sobel: la dérivation accentue le bruit de l'image, c'est-à-dire les pixels de valeur parasites et de répartition aléatoire. Les opérateurs de Sobel qui effectuent une moyenne locale sur 3 pixels en largeur sont moins sensibles aux bruits.

$$S_H = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \text{ Dérivation horizontale} \quad (1.21)$$

$$S_V = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \text{Dérivation verticale} \quad (1.22)$$

$$S_O = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix} \text{Dérivation oblique} \quad (1.23)$$

Dérivée secondes et Laplacien: Avec la même approximation que pour les dérivées premières on définit une dérivée partielle du second ordre suivant les colonnes de l'image par la matrice suivante :

$$D_{2H} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & -2 & 1 \\ 0 & 0 & 0 \end{bmatrix} \text{ Dérivée horizontale} \quad (1.24)$$

Suivant les lignes, la matrice s'écrit :

$$D_{2V} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -2 & 0 \\ 0 & 1 & 0 \end{bmatrix} \text{ Dérivée verticale} \qquad (1.25)$$

L'opérateur Laplacien donne une approximation directe de la somme de la dérivée seconde. Cet opérateur est utilisé pour détecter les contours d'une image. Le noyau du filtre est le suivant :

$$L = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \qquad (1.26)$$

*1.8.2.1.2. Utilisation de la transformé de Fourier*

Il est également possible de réaliser le filtrage d'une image dans le domaine fréquentielle. Pour cela, on multiplie la transformé de Fourier de l'image par la réponse fréquentielle du filtre et on calcule la T.F. inverse du résultat.

1.8.2.2. Filtrage non linéaire

Les filtres non linéaires ont été développés pour palier aux insuffisances des filtres linéaires principalement la mauvaise conservation des contours.

Une variété de filtre a été développée, qui ne sont pas linéaire. Un des filtres non linéaire le plus connu est *le filtre médian*.

Le filtre médian a été proposé par Tuckey en1970 pour l'analyse des séries temporelles. Il est capable de réduire certain type de bruit en dégradant très peu le contour, il a pour but de supprimer les bruits impulsionnels dans une image. Les bruits impulsionnels sont des points présents dans une image qui sont très différents de leur voisin.

*Principe* : pour supprimer ce type de bruits, on calcule l'intensité de la lumière sur les points voisins au point que l'on considère. Ensuite on trie les points en fonction de leur intensité lumineuse et on prend le point avec l'intensité médiane comme point résultant du filtrage.

*Illustration* :

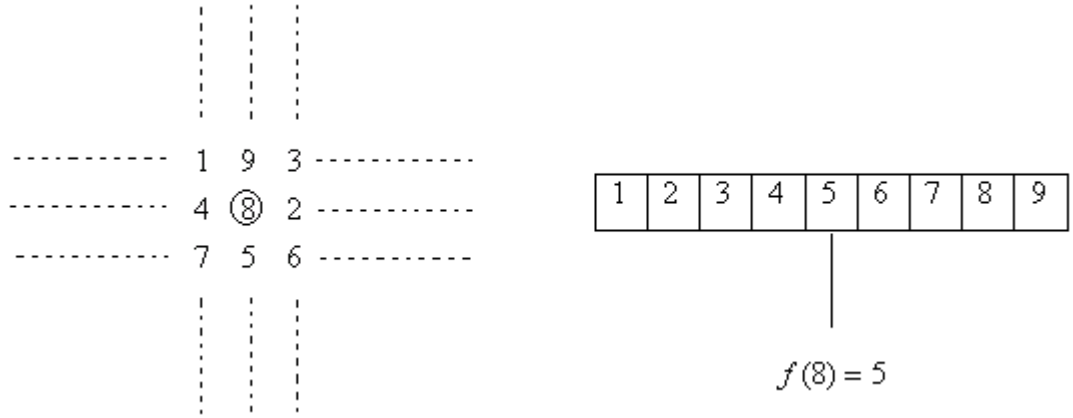


*Figure 1.10* : illustration de filtre médian

### *1.8.3. Analyse d'image*

L'analyse d'images a pour but l'extraction de l'information caractéristique contenue dans une image.

Le résultat d'une telle analyse s'appelle très souvent la description structurelle. Celle-ci peut prendre la forme d'une image ou de toute structure de données permettant une description des entités contenues dans l'image. Par opposition avec la phase d'interprétation, l'analyse tente, dans la mesure du possible, de ne pas prendre en compte le contexte attendu par les applications en avale. Essentiellement, l'analyse de l'image fait appel à la segmentation où l'on va tenter d'associer à chaque pixel de l'image un label en s'appuyant sur l'information portée (niveaux de gris ou couleur), sa distribution spatiale sur le support image, des modèles simples (le plus souvent des modèles géométriques).

La segmentation d'images ainsi définie est un domaine vaste où l'on retrouve de très nombreuses approches.

Toutes ces approches visent à l'extraction des indices visuels. Après de nombreuses années passées à rechercher la méthode optimale, les chercheurs ont compris que la segmentation idéale n'existait pas. On peut même montrer que le problème de la segmentation est le plus souvent un problème mal posé. Etant donnée une image, il existe toujours plusieurs segmentations possibles. Une bonne méthode de segmentation sera donc celle qui permettra d'arriver à une bonne interprétation. Elle devra donc avoir simplifié l'image sans pour autant en avoir trop réduit le

contenu. Entre autres, elle devra éviter les choix irréversibles. L'avenir de la segmentation est dans le pilotage aval par l'applicatif.

### 1.9. Conclusion

Nous avons vu dans cette partie des définitions concernant l'image comme ses propriétés physiques internes à savoir les intensités lumineuses, les niveaux de gris; ainsi que les traitements qu'elle peut subir.

Parmi ces traitements on a entamé préalablement aussi l’analyse en parlant un peu de la notion de segmentation qui consiste à préparer l'image pour les traitements hauts niveaux (interprétation de l'image), et la recherche des meilleurs moyens de le faire, qui est devenue une nécessité face à la croissance de la demande d'images et la complexité de celle-ci. Au cours du chapitre qui suit, nous présenterons différentes approches de segmentation applicable sur les images numériques.

# CHAPITRE 2
# ETAT DE L'ART DES TECHNIQUES DE SEGMENTATION D'IMAGES

## 2.1. Introduction

Le processus d'analyse d'images peut être défini comme l'ensemble de méthodes et d'outils permettant de décrire quantitativement le contenu d'une image. Il est généralement décomposé en plusieurs étapes: acquisition, numérisation, prétraitement, segmentation, interprétation.

L'étape de segmentation d'image est une étape très importante dans cette chaîne d'analyse car c'est à partir de l'image segmentée que les mesures sont effectuées pour l'extraction des paramètres discriminants en vue de la classification ou de l'interprétation. La tâche de segmentation peut se résumer de la manière suivante:

Etant donnée une image, l'objectif de la segmentation est d'établir une description compacte et représentative de son contenu informationnel, plus exploitable que l'ensemble de ses points. Il s'agit de procéder à l'extraction d'indices visuels (primitives) pertinents, suffisamment corrélés avec les entités qui composent la scène d'où l'image est prise [01][06][18][22].

La diversité des travaux menés dans le but de la compréhension de la vision humaine montre la complexité de la notion d'indice visuel et qu'il est bien difficile d'en donner une définition précise. Ceci étant, un indice visuel peut être défini comme une information perceptible directement à partir de la visualisation de l'image (contours, régions homogènes au sens des niveaux de gris et au sens de la texture, formes élémentaires). Ces indices visuels ou primitives correspondent à des photométriques et/ou géométriques remarquables [05][10].

Il existe dans la littérature une très grande diversité de techniques de segmentation. Nous pouvons en distinguer trois types de méthodes:

- Segmentation par seuillage,
- Segmentation par approche de région,
- Segmentation par détection de contour.

## 2.2. Segmentation par seuillage

Le seuillage a pour objectif de segmenter une image en plusieurs classes en n'utilisant que l'histogramme. On suppose donc que l'information associée à l'image permet à elle seule la segmentation, c'est-à-dire qu'une classe est caractérisée par sa distribution de niveaux de gris. Il

existe de très nombreuses méthodes de seuillages d'un histogramme. De plus, ces méthodes ont très souvent été développées pour traiter le cas particulier de la segmentation en deux classes (binarisation) et leur généralité face aux cas multiclasses n'est que très rarement garantie [17][25].
Le seuillage d'un histogramme en n classes consiste à déterminer n-1 seuils. Ce processus est appelé seuillage multi-niveaux (multilevel thresholding) tels que chaque classe soit associée à un intervalle de niveaux de gris distincts. Par exemple dans le cas le plus simple à 2 classes, le seuil s définit deux intervalles (deux classes) [0,s[ et [s,255].

En outre, le seuillage ne prend pas en considération les caractéristiques spatiales d'une image. Ceci la rend sensible aux bruits et aux inhomogénéités d'intensité. Pour ces raisons, des techniques de seuillage plus avancées ont été développées, on cite entre autres, le seuillage hystérésis, la détection des vallées, le seuillage entropique, la minimisation de la variance, maximisation du contraste, et la méthode du pourcentage [16][18].

D'une façon générale, la classification des points de l'image peut s'effectuer suivant les trois approches:

- approches globales,
- approches locales,
- approches adaptatives.

### *2.2.1. Seuillage par approche globale*

2.2.1.1. Principe

Le principe du seuillage global est d'utiliser des valeurs seuils à partir desquelles on peut choisir à quelle classe le pixel appartient. Si on préfère utiliser que deux classes, l'image segmentée sera une image binaire dont l'algorithme est le suivant :

| Algorithme 01 : binarisation |
| --- |
| si pixel > seuil alors<br>pixel =1<br>sinon<br>pixel =0<br>fin si |

La figure suivante explique mieux cet algorithme.

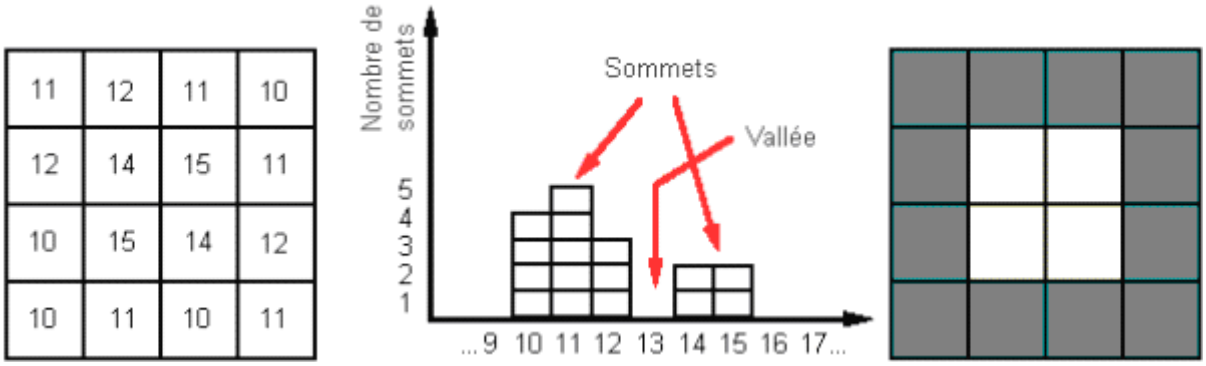


*Figure 2.01* : (a) matrice de l'image originale, (b) histogramme, (c) image binarisée

Dans ce cas, le seuil choisi est 13. Tous les niveaux de gris supérieurs à 13, on les affecte à une valeur un et les autres à zéro [12][23].

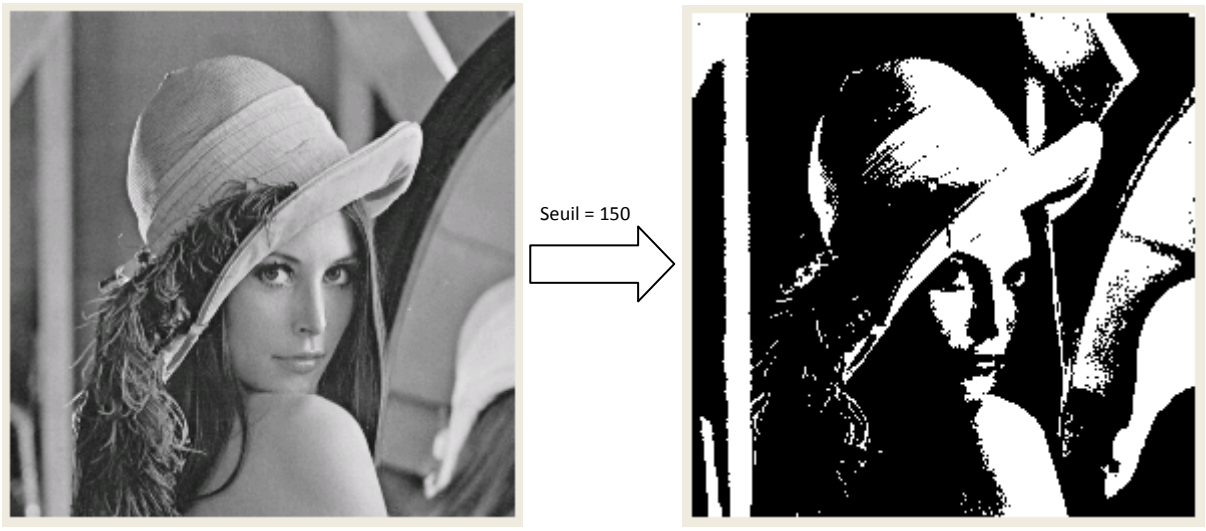


*Figure 2.02* : binarisation (algorithme 01)

Si le nombre de classe choisi est supérieur à deux, le premier algorithme n'est pas valable, d'où l'algorithme du seuillage multiple.

| Algorithme 02 : seuillage multiple |
|---|
| si le nombre de classe > 2, il faut définir n seuils pour $n+1$ classes<br>si pixel < seuil1 alors<br>pixel= p1<br>fin si<br>si pixel ≥ seuil1 et pixel < seuil2 alors<br>pixel= p2<br>fin si<br>…<br>si pixel ≥ seuiln alors<br>pixel= pn<br>fin si |

Illustrons cet algorithme par un exemple concret, on prendra quatre seuils: $S_1 = 50, S_2 = 100, S_3 = 150, S_4 = 200$.

$$pixel' = \begin{cases} 0 & si \quad pixel \prec 50 \\ 75 & si \quad 50 \le pixel \prec 100 \\ 125 & si \quad 100 \le pixel \prec 150 \\ 175 & si \quad 150 \le pixel \prec 200 \\ 255 & si \quad pixel \ge 200 \end{cases} \qquad (2.01)$$

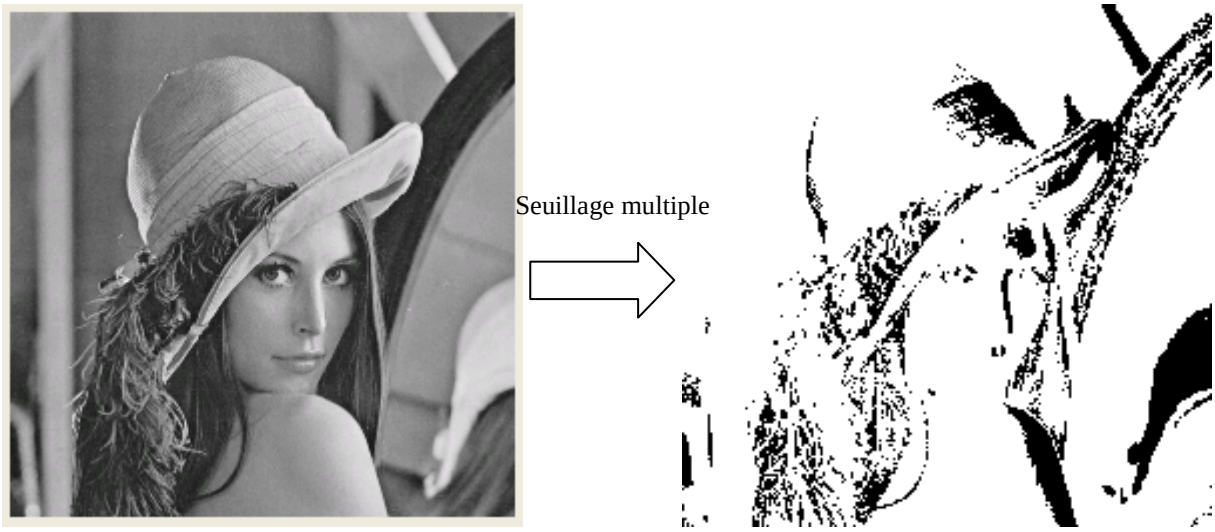


*Figure 2.03* : seuillage multiple (algorithme 02)

2.2.1.2. Choix du seuil

Avec cette approche, le problème est basé sur le choix du seuil pour la classification des niveaux de gris. En principe, il y deux méthodes dont la première est le choix du seuil manuel c'est-à-dire après avoir établi l'histogramme de l'image, il faut prélever les valeurs de niveau de gris qui se trouve dans les vallées. Quant à la deuxième méthode qui est le choix du seuil automatique, plusieurs algorithmes sont disponibles tel que l'algorithme d'OTSU né en 1979.

| Algorithme 03 : Otsu [05] |
| --- |
| On balaie toutes les valeurs possibles du seuil T<br>On calcule les moyennes et les variances de chaque classe<br>On s'intéresse à la variance intra-classes<br>Moyennes $\mu_1$ et $\mu_2$<br>Variances $\sigma_1^{\,2}$ et $\sigma_2^{\,2}$<br>Variance intra-classes : $\sigma w^2 = P_1\sigma_1^{\,2} + P_2\sigma_2^{\,2}$<br>Le seuil optimal est celui qui donne $\sigma w^2$ minimum |

Les formules suivantes représentent l'expression des variables mises en jeu dans l'algorithme :

$$\mu_1 = \frac{1}{T}\sum_{i=0}^{T-1} h(i)\,,\ P_1 = \frac{1}{MxN}\sum_{i=0}^{T-1} h(i) \qquad (2.02)$$

$$\mu_2 = \frac{1}{256-T}\sum_{i=T}^{255} h(i)\,,\ P_2 = \frac{1}{MxN}\sum_{i=T}^{255} h(i) \qquad (2.03)$$

$$\sigma_1^{\,2} = \frac{1}{T}\sum_{i=0}^{T-1}[h(i)-\mu_1]^2 \qquad (2.04)$$

$$\sigma_2^{\,2} = \frac{1}{256-T}\sum_{i=T}^{255}[h(i)-\mu_2]^2 \qquad (2.05)$$

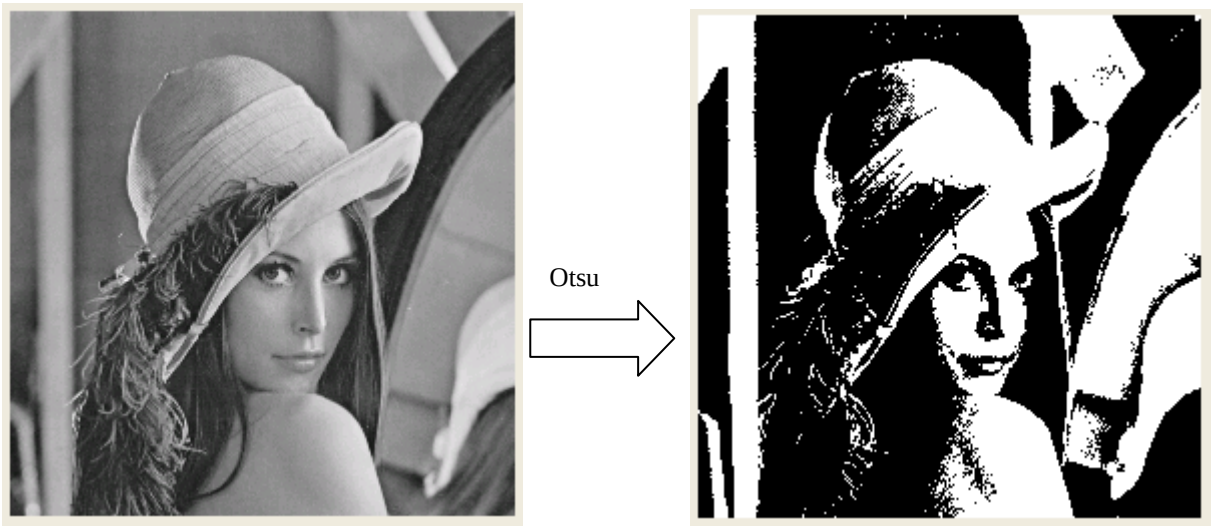


*Figure 2.04* : illustration de l'algorithme d'Otsu

2.2.1.3. Avantage et inconvénient du seuillage par approche global

L'avantage du seuillage par approche global est sa simplicité, en outre cette méthode s'applique correctement si l'histogramme contient des pics bien séparés. Par contre, cette segmentation n'est pas très utilisée en imagerie médicale puisque elle est affectée facilement par le bruit.

### *2.2.2. Seuillage par approche locale*

Le principe du seuillage local est d'utiliser une étude localisée autour du pixel pour déterminer quel seuil utiliser. Pour réaliser cette étude locale, les techniques utilisent une fenêtre d'étude centrée sur le pixel à étudier. Cette fenêtre peut avoir différentes tailles. Le premier à proposer une technique donnant de bons résultats fut Bernsen en 1986 [12]. Mathématiquement, le calcul du seuil peut s'écrire ainsi :

$$S(i, j) = (\max(i, j) + \min(i, j)) / 2 \qquad (2.07)$$

Avec :

- $S(i, j)$ : seuil à appliquer au point (i, j) ;
- max(i,j) : valeur du niveau de gris maximal dans une fenêtre centré en (i, j) de taille N×M ;
- min(i, j) : valeur du niveau de gris minimal dans une fenêtre centré en (i, j) de taille N ×M ;
- N et M sont des entiers naturels.

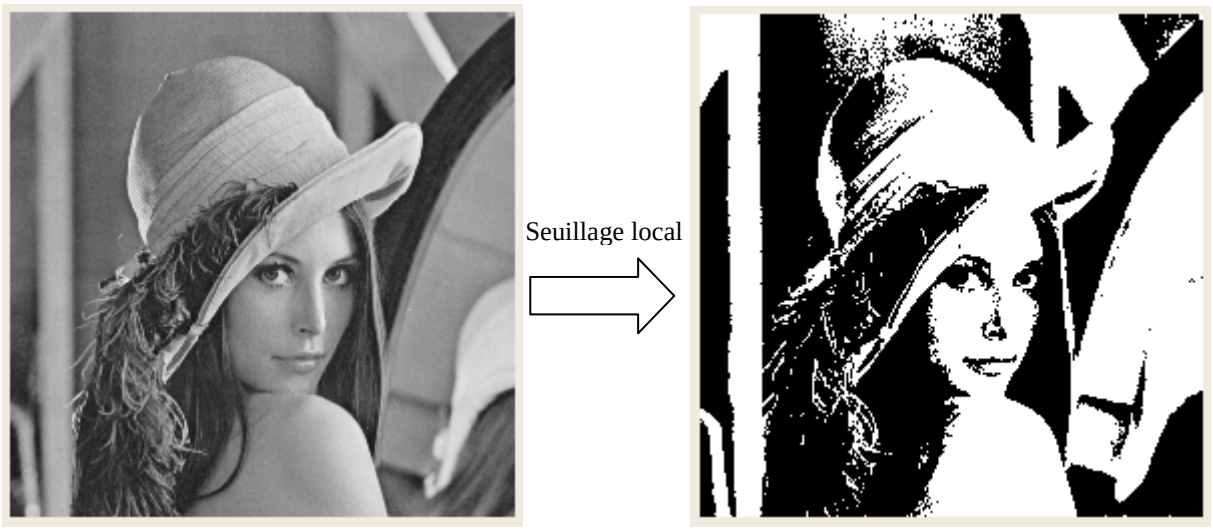


*Figure 2.05* : illustration du seuillage local

Cependant, ce filtre est très sensible au bruit du fond. A cause de la prise en compte du maximum et du minimum uniquement, dans le cas où la fenêtre est uniquement sur du fond, le bruit sera interprété comme objet, car le seuil sera bas. La même année, Niblack proposa une méthode similaire sur le principe, mais prenant en compte d'autres paramètres. Dans sa méthode, le seuil est calculé ainsi :

$$S(i, j) = \mu(i, j) + k\sigma(i, j) \qquad (2.08)$$

Avec :

- S(i, j) : seuil à appliquer au point (i, j) ;
- $\sigma$ (i, j) : valeur de l'écart type dans une fenêtre centré en (i, j) de taille N ×M ;
- μ(i, j) : valeur moyenne des niveaux de gris dans la même fenêtre ;
- k : constante fixée le plus généralement à 0,2 ;
- N et M sont des entiers naturels.

Bien que victime du même problème que précédemment, la méthode est plus efficace.

### *2.2.3. Seuillage par approche adaptative*

Dans le cas où le fond de l'image n'est pas uniforme (variation de luminosité), le seuillage global ne permet pas d'en extraire les objets. Pour pallier à cette difficulté, la solution la plus utilisée en pratique est basée sur l'application d'un seuillage adaptatif. Deux algorithmes sont plus connus à savoir : l'algorithme de k-means, et bayesienne [06].

2.2.3.1.Algorithme de k-means

L'algorithme des centres mobiles (k-means) est la technique de classification la plus simple. Cette technique utilise comme critère d'évaluation d'une partition la distance quadratique moyenne.

*2.2.3.1.1. Principe*

Soit un nuage de points (une image) que nous voulons partitionner en k-classes. Nous posons :

$x_j^{(i)}$ : le pixel j appartenant à la classe i,

$y_i$ : le centroïde de la classe i,

$L_i$ : le nombre de points de la classe i,

$d(x_j^{(i)}, y_i)$ : la distance (mesure de distorsion) entre $x_j^{(i)}$ et $y_i$ ; dans le cas de l'algorithme k-means, c'est la distance Euclidienne.

$D_i$ : la distance totale de la classe i,

$$D_i = \sum_{j=i}^{L_i} d(x_j^{(i)}, y_i) \qquad (2.09)$$

$D$ : la distorsion globale pour l'ensemble des vecteurs (inertie intraclasse),

$$D = \sum_{i=1}^{k} D_i \qquad (2.10)$$

La classification optimale est celle qui permet de minimiser la distorsion totale D. La procédure d'optimisation doit tenir compte des hypothèses suivantes :

- Pour un ensemble donné de centroïdes, la classification qui minimise *D* est celle pour laquelle chaque pixel est affecté à la classe dont le centroïde est le plus proche.
- Pour une classification donnée, il existe, pour chaque classe i, un pixel $y_i$ qui minimise la distance totale de la classe $D_i$.

| Algorithme 04: centres mobiles (k-means) [04] |
|---|
| Choisir un nombre de classes k.<br>Définir une classification C aléatoire : Choisir les k centroïdes $y_i$ de façon aléatoire dans l'espace des niveaux de gris [0, L −1].<br>Tant que l'inertie intraclasse n'est pas stable faire<br>Affecter chaque niveau de gris à la classe dont le centre est le plus proche.<br>Calculer les centres de gravité des classes de la nouvelle classification C' des classes.<br>C ←C'<br>Fin tant que<br>Afficher la classification obtenue. |

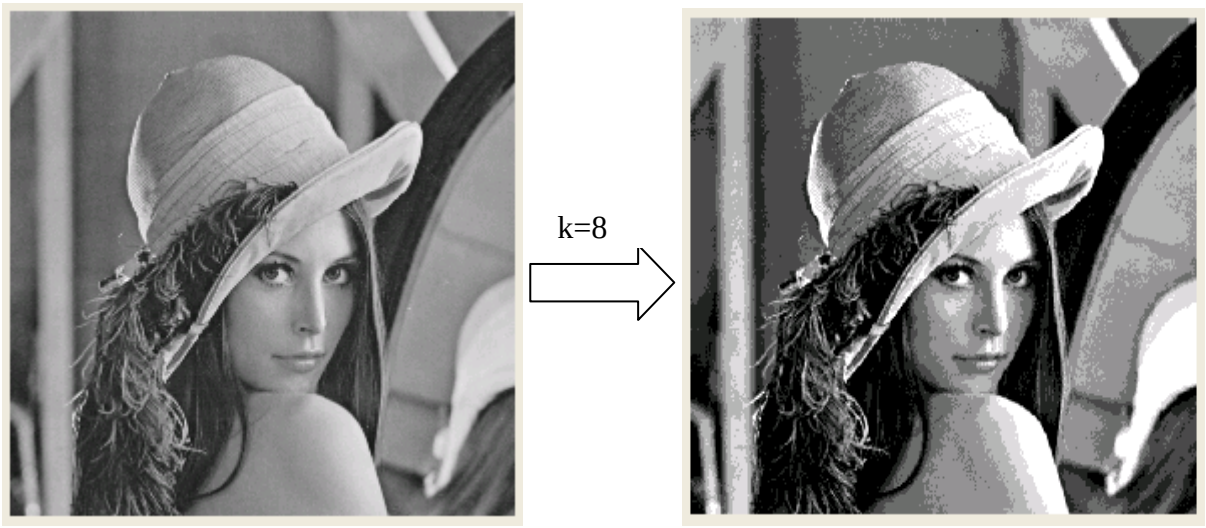


*Figure 2.06* : Illustration de l'algorithme de k-means

Le principal inconvénient de cette méthode est que la classification finale dépend du choix de la partition initiale. Le minimum global n'est pas obligatoirement atteint, on est seulement certain d'obtenir la meilleure partition à partir de la partition de départ choisie. La convergence de

l'algorithme est due au fait que $D_i$ décroit d'une itération à l'autre. De nombreuses variantes peuvent être rencontrées. Par exemple, au lieu de calculer le centre des classes, après avoir affecté tous les pixels, le centre de gravité peut être recalculé immédiatement après chaque affectation. La méthode des centres mobiles a été généralisée sous l'appellation de la « méthode des nuées dynamiques ». Au lieu de définir une classe par un seul point, son centre de gravité, elle est définie par un groupe de points (noyau de la classe).

### 2.3. Segmentation par région

Les approches régions divisent l'image en régions distinctes en recherchant non plus les seuils pour définir les classes comme dans l'approche de seuillage ci-précédente, mais la similarité des niveaux de gris des pixels en fonction d'un ou plusieurs attributs [03][15]. Parmi les techniques de segmentation en régions, on distingue usuellement quatre grandes classes d'algorithmes fonctionnant par :

- croissance de régions (region growing)
- division (split)
- fusion (merge)
- division/fusion (split and merge)

#### *2.3.1. Segmentation par croissance de régions (region growing)*

Les méthodes de croissance de régions utilisent la connexité des pixels pour former des régions homogènes. L'idée est de sélectionner, au début du traitement, plusieurs pixels qu'on appelle germes initiaux ensuite les pixels connexes à un germe respectant un certain critère de similarité (niveau de gris, ...) sont intégrés dans le germe. Ce dernier s'étend alors pour devenir une région. Ce processus est effectué pour tous les germes jusqu'à ce que tous les pixels soient traités. A la fin du traitement, l'image est découpée en un ensemble de régions issues de la croissance des germes.

Les avantages de cette technique [20] :

- Il s'agit d'une méthode simple et rapide.
- Elle permet la segmentation d'objet à topologie complexe.
- Elle préserve la forme de chaque région de l'image.

Les limites de cette technique :

- L'influence du choix des germes initiaux et le critère d'homogénéité sur le résultat de segmentation.
- Une mauvaise sélection des germes ou un choix du critère de similarité mal adapté peuvent entraîner des phénomènes de sous-segmentation (*intervient lorsqu'une région couvre plusieurs objets d'intérêt de classes différentes*) ou de sur-segmentation (*intervient quant les objets d'intérêt sont divisés en plusieurs régions à l'issus de la segmentation ce qui la rends de moins bonne qualité*).
- Il peut y avoir des pixels qui ne peuvent pas être classés.

2.3.1.1. Algorithme de croissance par régions

| Algorithme 05: croissance par régions [01] |
|---|
| Tant que l'image n'est pas segmentée en entier<br>{<br>Choisir un pixel non-étiqueté<br>Examiner les voisins<br>$v_j$ similaire $\Rightarrow$ étiquette $k$<br>Tant que $v_j \in$ région $k$<br>{<br>Examiner les voisins<br>$v_i$ similaire $\Rightarrow$ étiquette $k$<br>}<br>$k \leftarrow k+1$<br>} |

2.3.1.2. Illustrations

Avec l'exemple ci-dessous, on choisit au hasard quelques pixels de l'image. Dans notre cas, on prend le 10, ensuite il faut examiner ses 8 voisins (12, 11, 11, 10, 17, 18, 12,13). Le 13, 17, 18 ne respectent pas le seuil=2 et les autres sont étiquetés avec le 10. Puis, on procède à examiner les

voisins des pixels qui viennent d'être étiquetés  avec les même critères (seuil, voisin) et ainsi de suite. Les « ? » présentent les pixels non classés à la fin de la segmentation.

Image

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 32 | 15 | 13 | 11 | 12 | 10 | 13 | 9 |
| 12 | 12 | 11 | 11 | 13 | 11 | 8 | 10 |
| 15 | 13 | 10 | 10 | 12 | 14 | 16 | 9 |
| 14 | 12 | 18 | 17 | 11 | 14 | 20 | 19 |
| 13 | 11 | 16 | 17 | 9 | 11 | 18 | 17 |
| 13 | 11 | 16 | 17 | 9 | 11 | 17 | 19 |
| 12 | 10 | 18 | 16 | 10 | 11 | 16 | 20 |
| 15 | 12 | 17 | 19 | 11 | 10 | 18 | 22 |

Carte de régions (CR)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | 1 | 1 | 1 | | ? |
| | 1 | 1 | 1 | | 1 | ? | ? |
| | | 1 | 1 | 1 | | | ? |
| | 1 | | | 1 | | 2 | 2 |
| | 1 | | | 1 | 1 | 2 | 2 |
| | 1 | | | 1 | 1 | 2 | 2 |
| 1 | 1 | | | 1 | 1 | 2 | 2 |
| | 1 | | | 1 | 1 | 2 | |

*Figure 2.07* : illustration de la technique par croissance région avec seuil=2 et 8-voisins

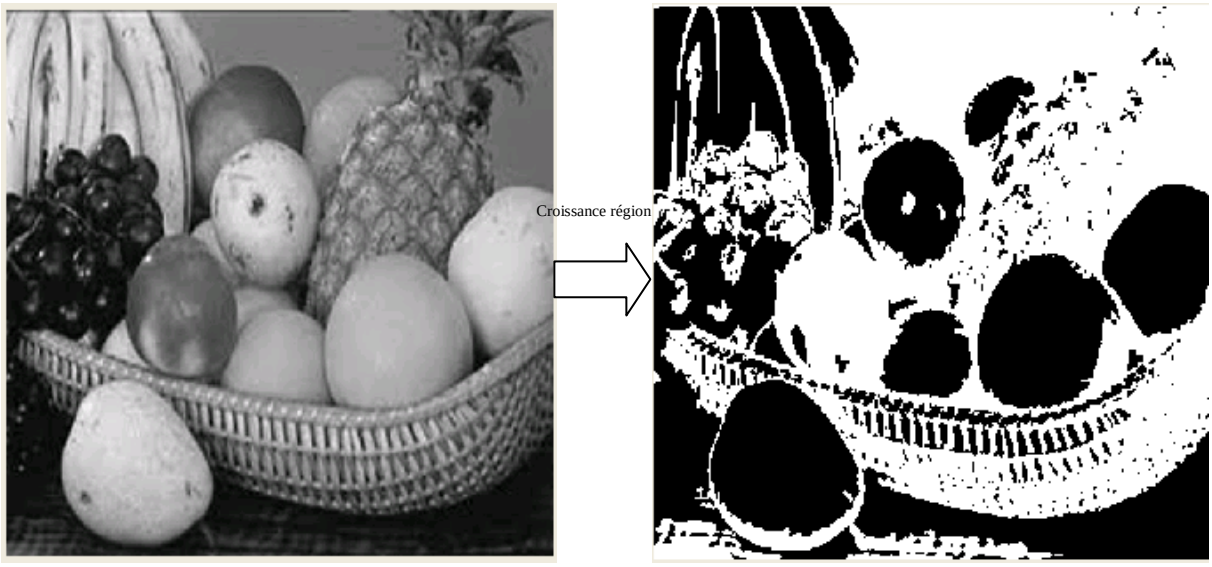


*Figure 2.08* : segmentation par croissance régions

### *2.3.2. Segmentation par division de régions (split)*

Le principe consiste à tester d'abord le critère d'homogénéité retenu sur l'image entière.

Si le critère est valide, l'image est considérée comme segmentée ; sinon, l'image est découpée en zones plus petites et la méthode est réappliquée sur chacune des zones. La division peut se faire en quatre parties, en six parties, en polygones, etc. L'approche *quadtree* permette d'effectuer cette approche [20].

2.3.2.1. Tetra-arbre (quadtree)

Le quadtree est un tétra-arbre dont chaque nœud possède exactement quatre nœuds fils, excepté les nœuds terminaux. Chaque nœud correspond à un bloc, c'est à dire une zone de l'image de forme carrée. La racine de l'arbre correspond à l'image entière.

Chaque bloc associé à un nœud du quadtree de la partition initiale est analysé de façon récursive afin de décider s'il doit être divisé en quatre sous-blocs. L'analyse récursive s'arrête lorsque chaque sous-bloc respecte un prédicat d'homogénéité [11].

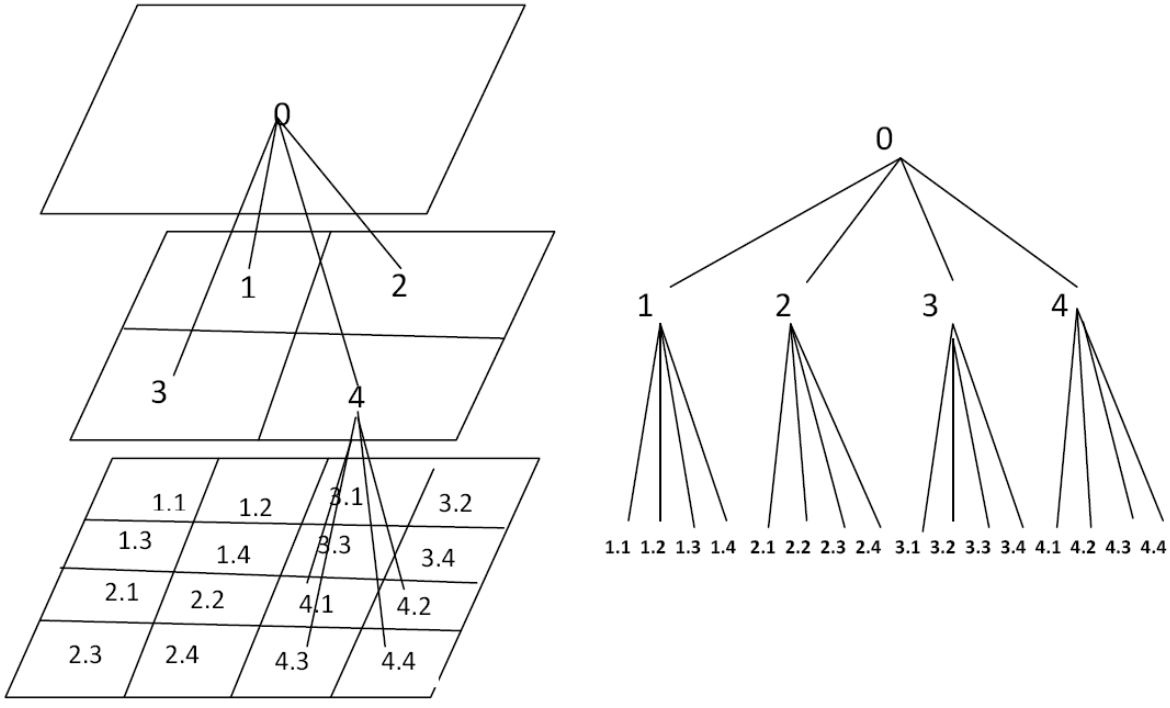


*Figure 2.09* : structure pyramidale du quadtree

Les principales étapes de l’algorithme de division par quadtree sont les suivants :

| Algorithme 06: division par régions |
| --- |
| La racine de l’arbre correspond à l’image entière (le premier bloc)<br>si tous les blocs respectent le prédicat d’homogénéité<br>Segmentation terminée<br>sinon<br>Diviser chaque bloc qui ne respecte pas le critère d’homogénéité en quatre sous- blocs.<br>fin si |

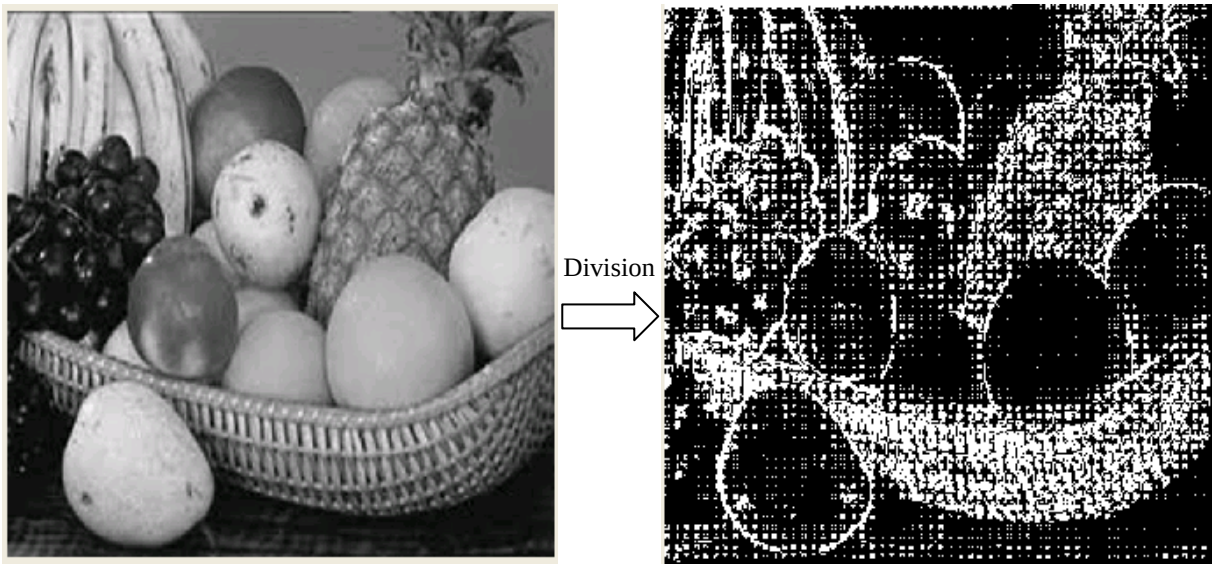


*Figure 2.10* : segmentation par division de régions

2.3.2.2. Inconvénients

- La rigidité du découpage carré qu’il impose.
- Il conduit à un partitionnement global de l’image qui ne respecte pas toujours la forme des régions présentes dans l’image.
- La division par quadtree fournit généralement une sur-segmentation car elle décompose le bloc en 4 sous-blocs en cas de non-homogénéité. Cependant, dans les dernières itérations, la décomposition en 4 sous-blocs n'est normalement pas nécessaire. En effet, parmi les 4 sous-blocs créés, plusieurs sont similaires.

### *2.3.3. Segmentation par fusion de régions (merge)*

L'approche de fusion de régions est principalement fondée sur l'analyse d'un graphe d'adjacence de régions qui traite une image pré-segmentée, constituée d'un ensemble de régions.

Le graphe d'adjacence est une structure de données constituée d'un graphe non-orienté dont chaque nœud représente une région et chaque arête représente une adjacence (la différence de niveau de gris moyen, la différence de nombre de région,…) entre deux régions [20][21].

La figure 2.11 représente, à titre d'exemple, une image et son graphe d'adjacence de régions.

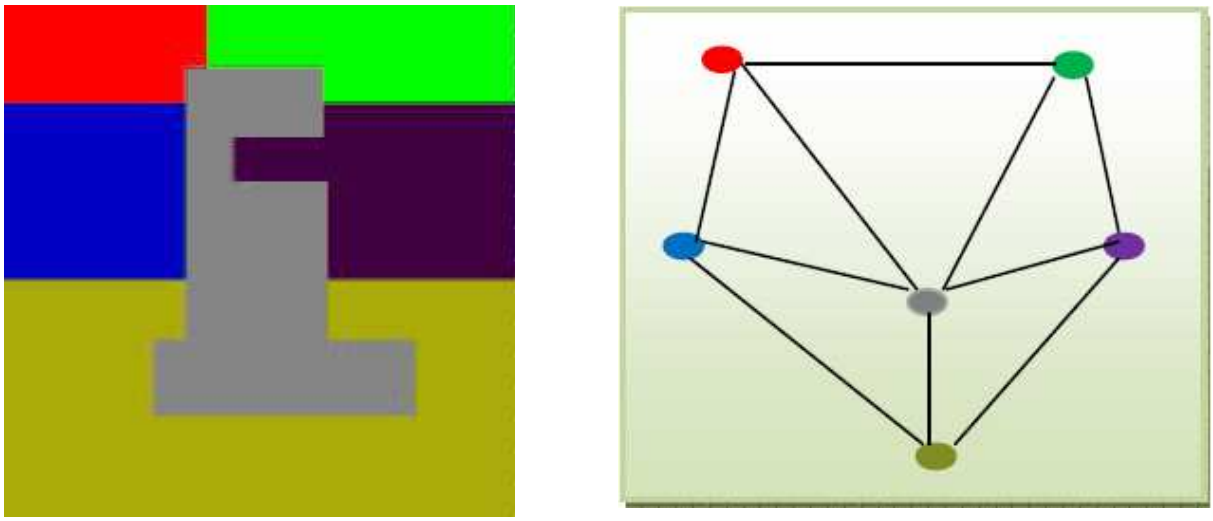

*Figure 2.11* : image et son graphe d'adjacence

L'analyse du graphe d'adjacence de régions permet de fusionner des régions d'une image sur-segmentée. Le procédé consiste à fusionner deux nœuds reliés par une arête à condition qu'ils respectent un critère de fusion. Les méthodes d'analyse des graphes d'adjacence de régions se distinguent selon l'ordre de parcours des différents nœuds du graphe et selon les critères de fusion. A chaque itération, les régions reliées par l'arête qui porte la valeur minimale sont fusionnées. Les valeurs des arêtes sont mises à jour en fonction du nombre de pixels appartenant aux régions associées aux arêtes et de leur distance colorimétrique.

L'algorithme de fusion s'arrête lorsqu'un nombre d'itérations fixé préalablement est atteint ou lorsque les poids des arêtes atteignent une valeur limite.

| Algorithme 07: fusion de régions |
| --- |
| Tant qu'un critère de fusion est satisfait<br>{<br>Chercher l'arête *A* qui porte la valeur minimale<br>Fusionner les deux sommets de *A* (les deux régions les plus semblables)<br>Supprimer l'arête *A*<br>Mettre à jour le graphe<br>} |

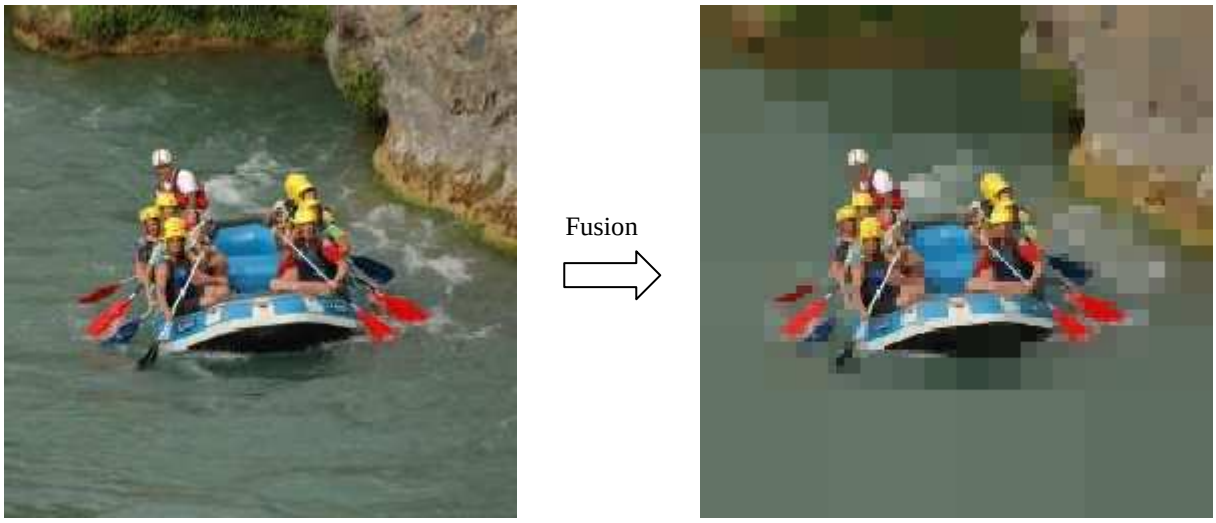


*Figure 2.12* : image segmentée par fusion

Les inconvénients de cette méthode se situent à deux niveaux :

- Cette méthode dépend du critère de fusion qui peut influer sur le résultat final de la segmentation,
- Elle peut introduire l'effet de sous-segmentation.

### *2.3.4. Segmentation par dual division – fusion (split & merge)*

La segmentation par division-fusion qui a été proposée par HOROWITZ et PAVLIDIS en 1974 regroupe les deux types d'approches précédentes [20].

Tout d'abord, l'image est divisée en régions homogènes, puis les régions adjacentes qui répondent à des critères de fusion sont fusionnées. Voici un exemple de segmentation par division/fusion.

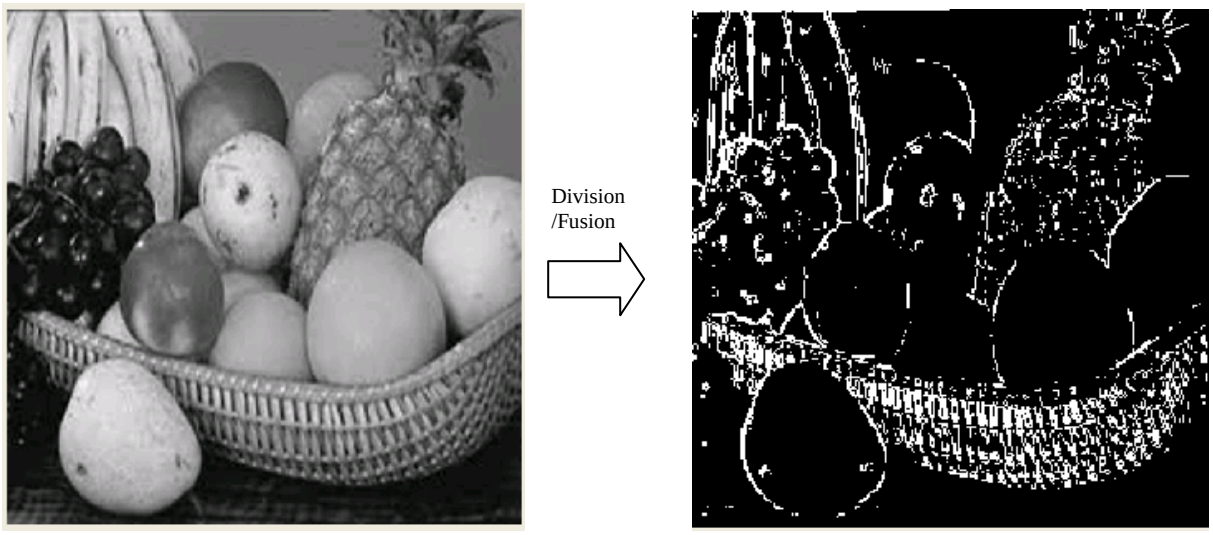


*Figure 2.13* : image segmentée par division/fusion

### *2.3.5. Inconvénients de la segmentation par région*

Les inconvénients de la segmentation par région se situent à trois niveaux [20]:

- Les régions obtenues ne correspondent pas, dans tous les cas, aux objets représentés dans l'image.
- Les limites des régions obtenues sont habituellement imprécises et ne coïncident pas exactement aux limites des objets de l'image.
- La difficulté d'identifier les critères pour agréger les pixels ou pour fusionner et diviser les régions.

## 2.4. Segmentation par contours

Un contour peut être défini comme étant la limite ou la frontière entre régions adjacentes dans une image. Les méthodes orientées contours visent à délimiter les objets de l'image selon leurs contours. Elles se basent pour cela sur la variation d'intensité entre les pixels de l'image, une variation ou un changement d'intensité brusque se traduit par une frontière entre régions voisines.
L'approche contours consiste à identifier les transitions qui existent entre les régions.
Cependant, les discontinuités dans l'image ne sont pas dues uniquement aux différentes structures de l'image, elles peuvent être provoquées par une différence d'éclairage, par exemple un effet d'ombre. C'est la raison pour laquelle les contours détectés ne sont pas toujours connexes. Pour

remédier à ce problème, des techniques ont été proposées afin d'obtenir des contours fermes tel que le filtre de *Canny* et le filtre de *Deriche* [22].
Il existe plusieurs méthodes utilisant l'approche contours, dans ce qui suit, nous allons citer les plus importantes.

### *2.4.1. Les approches dérivatives*

Pour déterminer la position des éventuels contours on utilise les méthodes dérivatives. L'image dérivée met en évidence les variations de niveau de gris. L'approche dérivative consiste à balayer une image avec une fenêtre définissant la zone d'intérêt. A chaque position, un opérateur est appliqué sur les pixels de la fenêtre afin d'estimer s'il y a une transition significative au niveau de l'attribut choisi. Le résultat obtenu sera alors une image binaire constituée de deux classes: les pixels des contours et les pixels des non-contours. A partir des pixels susceptibles d'appartenir à un contour, il faut ensuite extraire des contours fermés [15].
On peut classer les méthodes dérivatives selon deux approches:

- Approche Gradient.
- Approche Laplacien.

#### 2.4.1.1. Le gradient d'une image

Le gradient, en un pixel d'une image numérique, est un vecteur caractérisé par son amplitude et sa direction. L'amplitude est directement liée à la quantité de variation locale des niveaux de gris. La direction du gradient est orthogonale à la frontière qui passe au point considéré.

Soit $f(x,y)$ une fonction continue qui représente l'intensité de chaque point de l'image I, le gradient de I en un point est le vecteur $\nabla f(x,y)$ définit comme suit:

$$\nabla f(x,y) = \begin{bmatrix} \dfrac{\partial f(x,y)}{\partial x} \\ \dfrac{\partial f(x,y)}{\partial y} \end{bmatrix} \qquad (2.11)$$

Pour calculer ce gradient en chaque point de I, on effectue, généralement, le produit de convolution de I avec un opérateur de dérivation fournissant deux masques $M_x$ et $M_y$ correspondant aux directions verticale et horizontale tels que:

$$\frac{\partial f(x,y)}{\partial x} = G_x(x,y) = M_x * f(x,y) \qquad (2.12)$$

Et

$$\frac{\partial f(x,y)}{\partial y} = G_y(x,y) = M_y * f(x,y) \qquad (2.13)$$

L'amplitude du gradient s'obtient par la formule suivante :

$$G(x,y) = \sqrt{G_x(x,y)^2 + G_y(x,y)^2} \approx \left|G_x(x,y)\right| + \left|G_y(x,y)\right| \qquad (2.14)$$

Et la direction du gradient est donnée par :

$$d(x,y) = \arctan\frac{G_y(x,y)}{G_x(x,y)} \qquad (2.15)$$

| *NOM* | $M_x$ | $M_y$ |
|---|---|---|
| Roberts | $\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$ | $\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}$ |
| Prewitt | $\frac{1}{3}\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$ | $\frac{1}{3}\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$ |
| Sobel | $\frac{1}{4}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$ | $\frac{1}{4}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$ |

*Tableau 2.01* : principaux masques 3 x 3 de calcul de gradient

Afin de calculer les dérivées, il existe de très nombreuses opérations différentes, les plus célèbres sont regroupées au tableau ci-dessus.

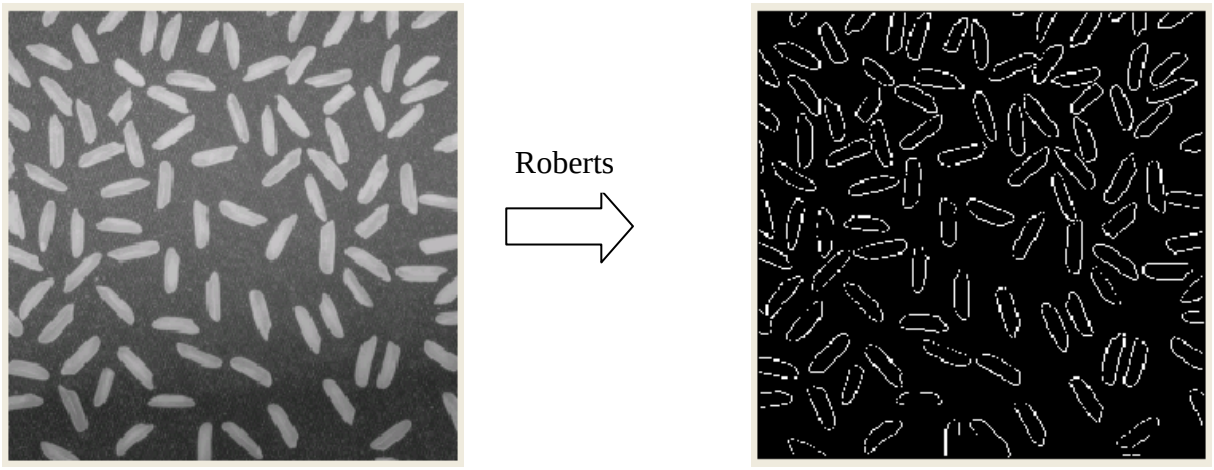


*Figure 2.14* : image segmentée par le masque de Roberts

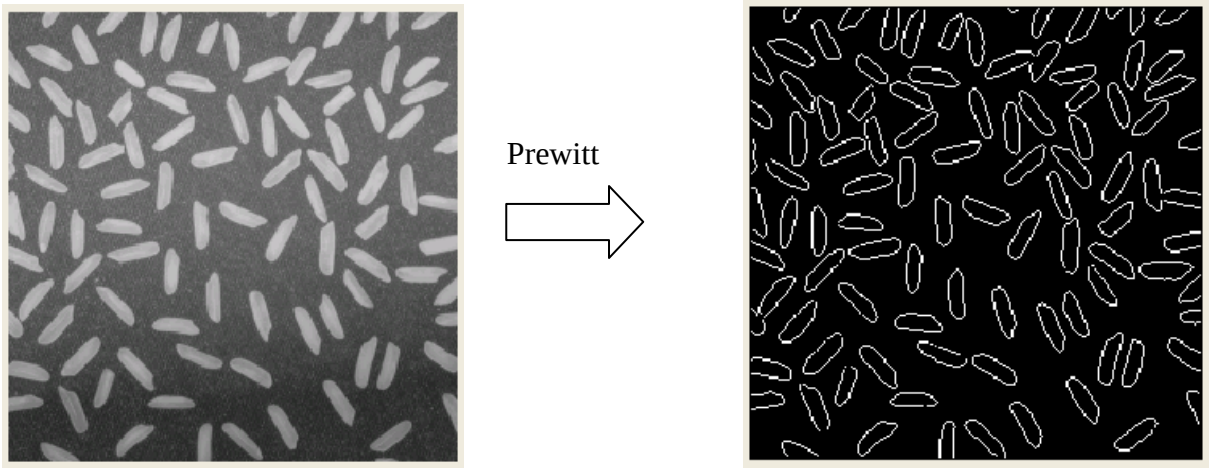


*Figure 2.15* : image segmentée par le masque de Prewitt

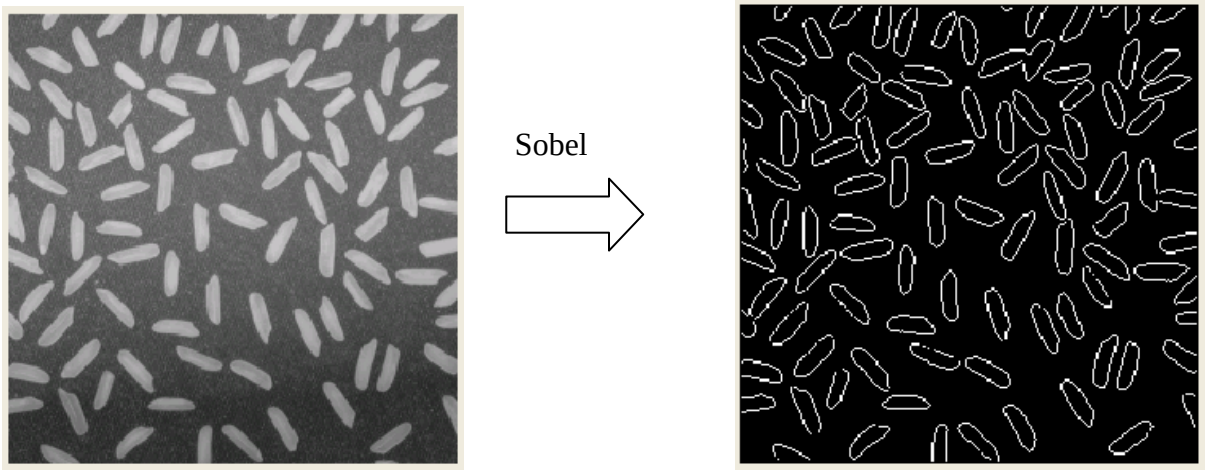


*Figure 2.16* : image segmentée par le masque de Sobel

Dans le cas d'une approche dérivée première, on dispose donc de la valeur gradient en tout point de l'image soit la fonction G. L'extraction des points de contour s'effectuait par sélection des points de norme de gradient élevé grâce aux deux étapes suivantes:

- Calcul de la norme du gradient:

$$N(x,y) = \sqrt{G_x(x,y)^2 + G_y(x,y)^2}$$

- Sélection des points de fort gradient: on détermine les points tels que $G(x,y) > s$ avec s : seuil fixe priori.

Dans une deuxième étape on élimine les points de norme de gradient faible avec un seuillage. Ce type de seuillage permet l'obtention de points de contour bien connexes entre eux.

2.4.1.2. Laplacien d'une image

Laplacien en un pixel d'une image numérique est calculé par la relation suivante:

$$L(x,y) = \frac{\partial^2 f(x,y)}{\partial^2 x} + \frac{\partial^2 f(x,y)}{\partial^2 y} \qquad (2.16)$$

Pour calculer laplacien en chaque point de l'image, on effectue, généralement, le produit de convolution de l'image avec un masque.

Nous verrons ci-dessous quelques exemples de masque :

$$M_1 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \qquad (2.17)$$

$$M_2 = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \qquad (2.18)$$

$$M_3 = \begin{bmatrix} -1 & 0 & 0 & -1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ -1 & 0 & 0 & -1 \end{bmatrix} \qquad (2.19)$$

Dans le cas d'une approche dérivée seconde, on dispose dans la valeur de laplacien en chaque point de l'image soit la fonction L. On considère que les points de contours sont localisés aux passages par zéro du laplacien. Si le calcul de laplacien était exact il suffirait de sélectionner les points M tels que $L(M) = 0$, mais comme généralement l'approximation de laplacien est assez bruitée, on détecte les points où il change de signe. Une dernière étape du seuillage est là encore nécessaire afin d'éliminer les points de trop faible gradient.
L'extraction de ces passages par zéro s'effectue classiquement en trois étapes:

- Détermination d'une image de polarité: on calcule une image $I_p$ telle que :

$$I_p(M) = \begin{cases} 0 \, si & L(M) > 0 \\ 1 \, si & L(M) \leq 0 \end{cases} \qquad (2.20)$$

- Détection des passages par zéro: on calcule une image $I_z$ telle que :

$$I_z(M) = \begin{cases} 1 \quad si & \text{M correspond à une transition } 0\text{-}1 \text{ ou } 1\text{-}0 \text{ dans } I_p \\ 0 \, ailleurs \end{cases} \qquad (2.21)$$

- Seuillage des passages par zéro: l'élimination des passages par zéro de faible norme de gradient peut s'effectuer par un algorithme de seuillage quelconque.

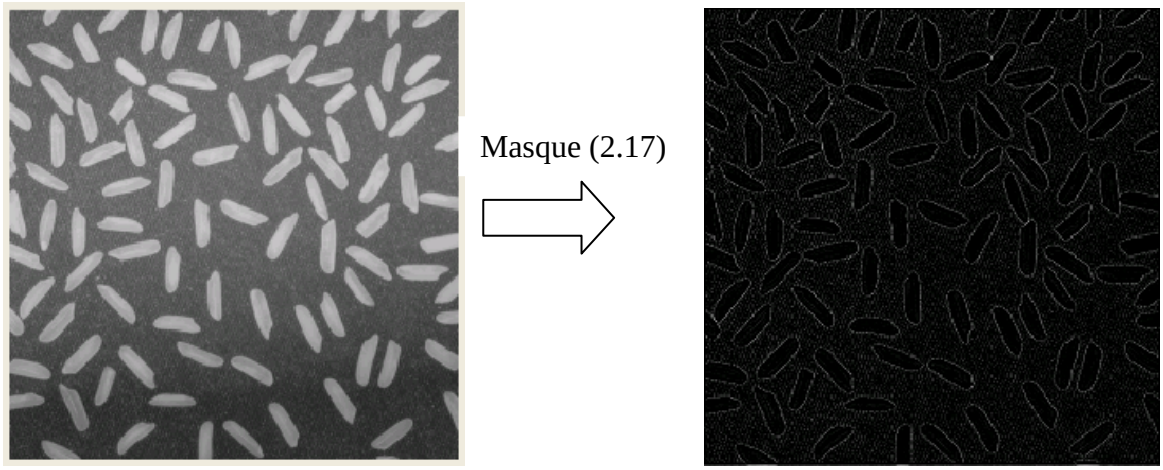


*Figure 2.17* : image segmentée par le masque 2.17

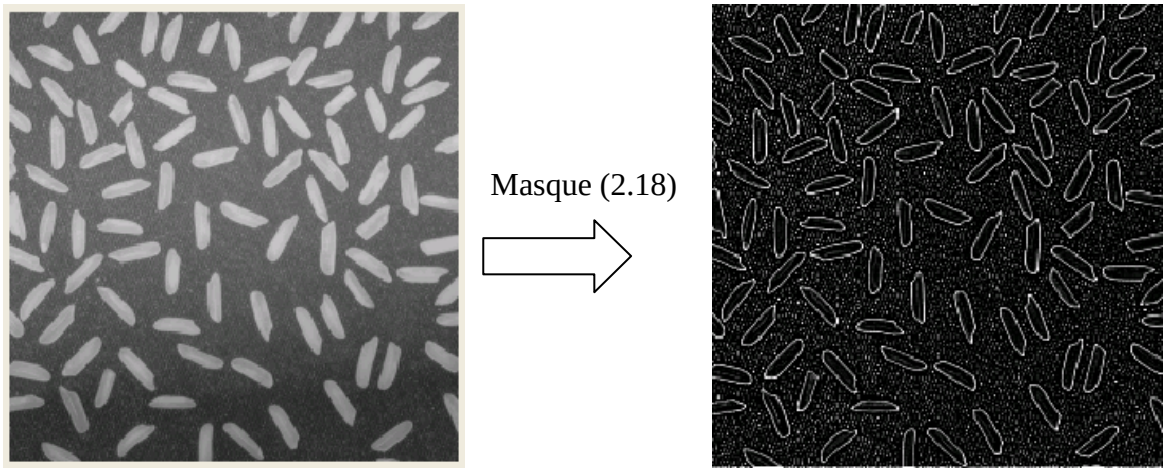


*Figure 2.18* : image segmentée par le masque 2.18

2.4.1.3. Les filtres sophistiqués

Pour améliorer la qualité des contours et pallier aux problèmes de précision de localisation et d'efficacité de détection, les filtres sophistiqués sont apparus. Ces filtres sont des opérateurs de dérivation avec filtrages optimaux. Le filtre optimal est un dérivateur qui permet de détecter des contours en respectant les 3 critères suivants [20] :

- Une bonne détection de contour.
- Une bonne localisation : optimisation de la précision avec laquelle le contour est détecté.
- Unicité de la réponse : le contour provoque une réponse unique de l'opérateur.

Parmi ces operateurs, on trouve : le filtre de *Canny*, le filtre de *Deriche* et le filtre de *Shen et Castan*

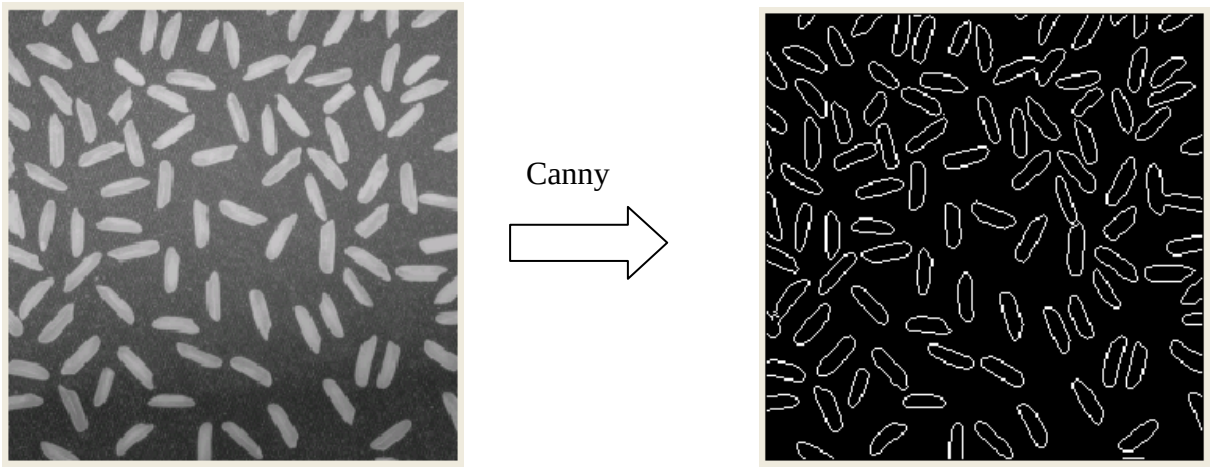


*Figure 2.19* : image filtrée par le filtre de Canny

Les méthodes dérivatives ne nécessitent pas d'information a priori, elles ne permettent pas, en général, d'obtenir des contours fermés. Elles sont en effet très sensibles au bruit.

### *2.4.2. Modèles déformables (contour actif)*

Les algorithmes de segmentation fondés sur les modèles déformables ont l'avantage, par rapport aux méthodes dérivatives, de fournir des contours ou surfaces fermés. Ces méthodes sont connues sous le nom du « contour actif » ou « snake ».

Un contour actif est défini comme une courbe minimisant une énergie et évoluant de manière itérative à partir d'une position initiale proche du contour recherché jusqu'à convergence

(Obtention de contour recherché)

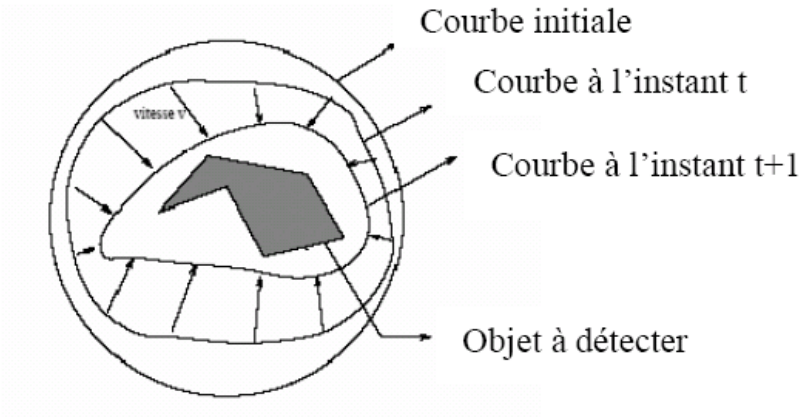


*Figure 2.20* : évolution du contour actif

Le contour est représenté par une courbe $C = v(s,t)$, ouverte ou fermée paramétrée par l'abscisse curviligne s tel que $s \in [a,b]$ et le temps t. Le processus de déformation est lié à la minimisation d'une fonctionnelle d'énergie, notée E et composée de deux termes

$$E(c) = E_{\text{int}}(c) + E_{ext}(c) \qquad (2.22)$$

Où $E_{\text{int}}(c)$ représente l'énergie interne et $E_{ext}(c)$ représente l'énergie externe.

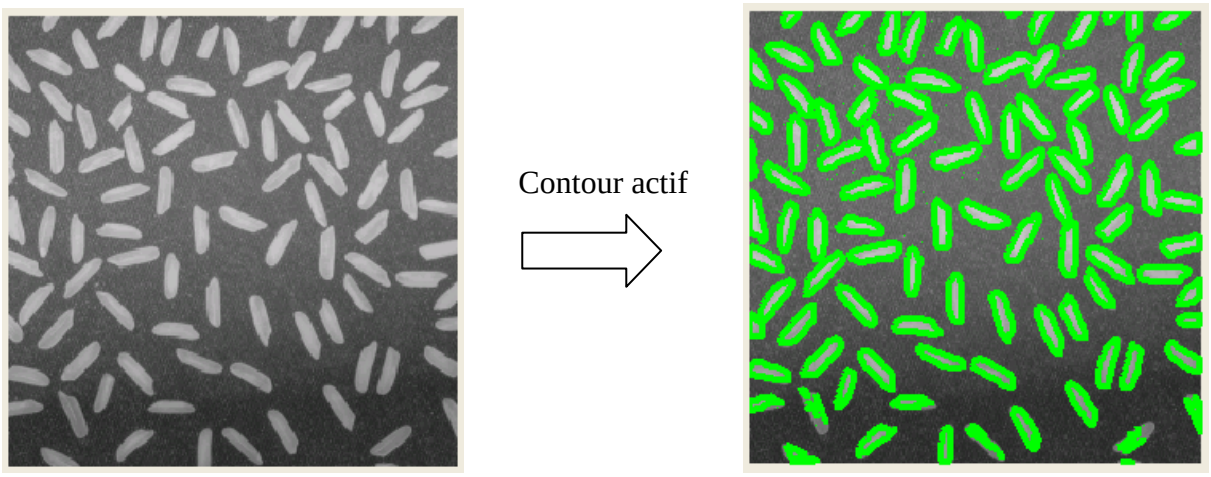


*Figure 2.21* : image segmentée par contour actif

2.4.2.1. Energie interne

Appelée aussi énergie géométrique, elle contrôle et régularise l'aspect de la courbe, cette énergie assure au contour une certaine continuité atténuant les effets du bruit [11]. Sa formule générale est donnée comme ci-dessous :

$$E_{\text{int}}(c) = \sum_{r=0}^{p} \int_{a}^{b} \alpha_r(s) \left| \frac{\partial^r v(s)}{\partial s^r} \right|^2 ds \qquad (2.23)$$

Où $v(s) = (x(s), y(s))$ est le point courant du contour $C$

Kass et Al se sont limités dans leur modèle au cas où $p = 2$ et où les coefficients $\alpha_r(s)$ sont constants vis-à-vis de la variable s, l'énergie interne devient alors :

$$E_{\text{int}}(c) = \int_{a}^{b} \alpha(s) \left| \frac{\partial v(s)}{\partial s} \right|^2 ds + \int_{a}^{b} \beta(s) \left| \frac{\partial^2 v(s)}{\partial s^2} \right|^2 ds \qquad (2.24)$$

Tel que : le premier terme mesure l'élongation totale de la courbe et il est lié à sa rigidité alors que le deuxième terme la courbure et il est lié à l'élasticité [15].

2.4.2.2. Energie externe

Appelée aussi énergie image. Elle permet d'attirer la courbe vers l'objet dont on cherche les frontières. Cette énergie fait intervenir la caractéristique image que l'on cherche à mettre en valeur. En effet, pour mettre en valeur les zones de fort contraste, nous pouvons choisir une énergie image donnée par la relation :

$$E_{\text{int}}(c) = -\int_a^b \left|\nabla^* I(v(s))\right|^2 ds \qquad (2.25)$$

Tel que : $\nabla^* I(v(s))$ est le gradient de l'intensité de l'image I au voisinage de la courbe $v(s)$ [15]

## 2.5. Evaluation de la segmentation

Beaucoup de travaux ont été réalisés dans le domaine de la segmentation d'images, mais malheureusement, rares sont les personnes qui ont évalué leurs algorithmes. Dans la plupart des cas, ils s'arrêtent à un résultat visuel alors que ce n'est pas une preuve de l'efficacité de la segmentation.

Pour évaluer une segmentation, il existe des bases d'images synthétiques dont la segmentation est connue. Nous pouvons ainsi avoir un modèle de comparaison pour notre résultat de segmentation. Parmi ces bases d'images, nous pouvons citer la base d'images Brainweb.

La comparaison se fait selon plusieurs critères :

### *2.5.1. Calcul du pourcentage de pixels mal classés*

Le calcul du pourcentage de pixels mal classés est donné par la formule suivante :

$$\alpha_{mal\_classés} = \frac{NPMC}{NTP} \times 100 \qquad (2.26)$$

Plus $\alpha_{mal\_classés}$ est petit, plus la segmentation est bonne [22].

### *2.5.2. Calcul du pourcentage de pixels bien classés*

Il est calculé à partir de l'expression suivante :

$$\alpha_{bien\_classés} = \frac{NPBC}{NTP} \times 100 \qquad (2.27)$$

Plus $\alpha_{bien\_classés}$ est grand, plus la segmentation est bonne [22].

### *2.5.3. Le coefficient de Dice*

Le coefficient de Dice ou encore appelé Kappa Index permet de mesurer la ressemblance entre 2 régions. Notons $r$ une région issue du résultat de segmentation et $r^*$ la même région dans la vraie segmentation :

$$KI = \frac{2TP}{2TP + FP + FN} \tag{2.28}$$

Où : TP (True Positive) : nombre de pixels présents dans $r^*$ et présents dans $r$

FP (False Positive) : nombre de pixels présents dans $r^*$ et absents dans $r$

FN (False Negative) : nombre de pixels absents dans $r^*$ et présents dans $r$

Le coefficient de Dice est entre 0 et 1. Lorsqu'il est égal à 1, la segmentation est parfaite [22].

## 2.6. Etude comparatif des techniques de segmentation

### *2.6.1. Tableau comparatif*

Le tableau suivant présume les avantages et inconvénients de chacune des ses méthodes étudiées dans les paragraphes précédents.

| ***Segmentation par approche de :*** | ***AVANTAGES*** | ***INCONVENIENTS*** |
|---|---|---|
| Seuillage | ▪ Rapidité de la segmentation<br>▪ Facilité de la technique | ▪ Difficultés sur la détermination des seuils<br>▪ Elle ne tient pas compte la position des pixels de l'image |
| Régions | ▪ La fermeture des contours<br>▪ La densité de l'information extraite [09] | ▪ Les régions ne correspondent pas toujours aux objets contenus dans l'image<br>▪ Les limites des régions sont généralement imprécises |
| Contours | ▪ Rapidité de la segmentation.<br>▪ Précision de la segmentation | ▪ Existence de faux contours<br>▪ Les contours ne sont généralement pas fermés [19]. |

*Tableau 2.02* : comparaison des techniques de segmentation

### *2.6.2. Mesure de performance*

En appliquant le critère du pourcentage de pixels mal classé (2.26) et en prenant une image de référence dans la base de Brainweb (figure 2.02) comme paramètre d'entrée, les trois graphes qui suivent, présentent la mesure de performance des algorithmes vus tout au long de ce second chapitre.

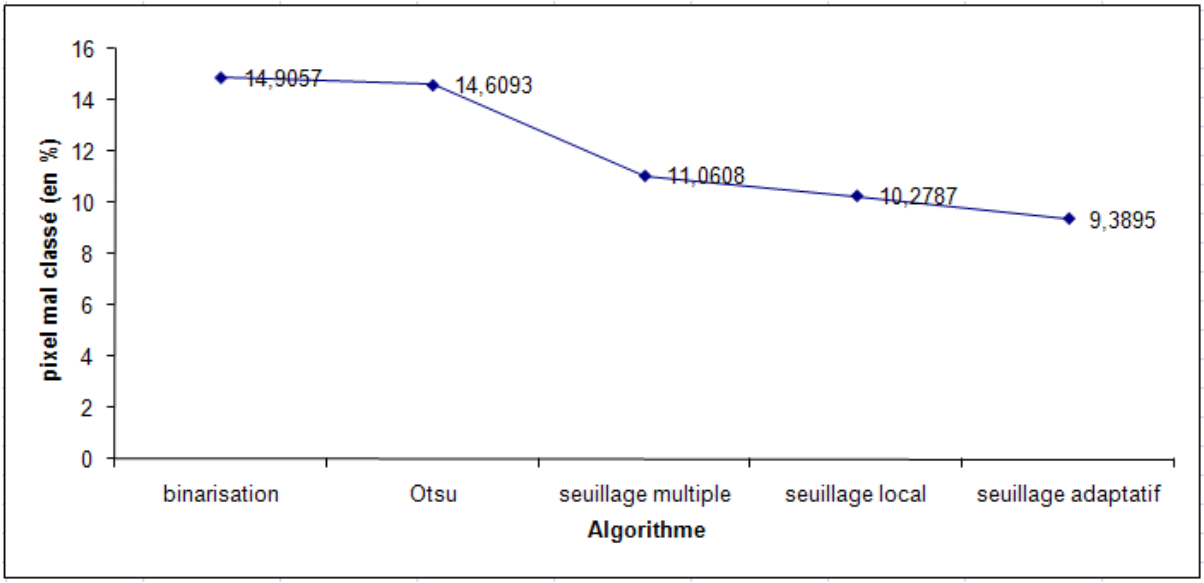


*Figure 2.22* : mesure de performance des algorithmes de segmentation par seuillage

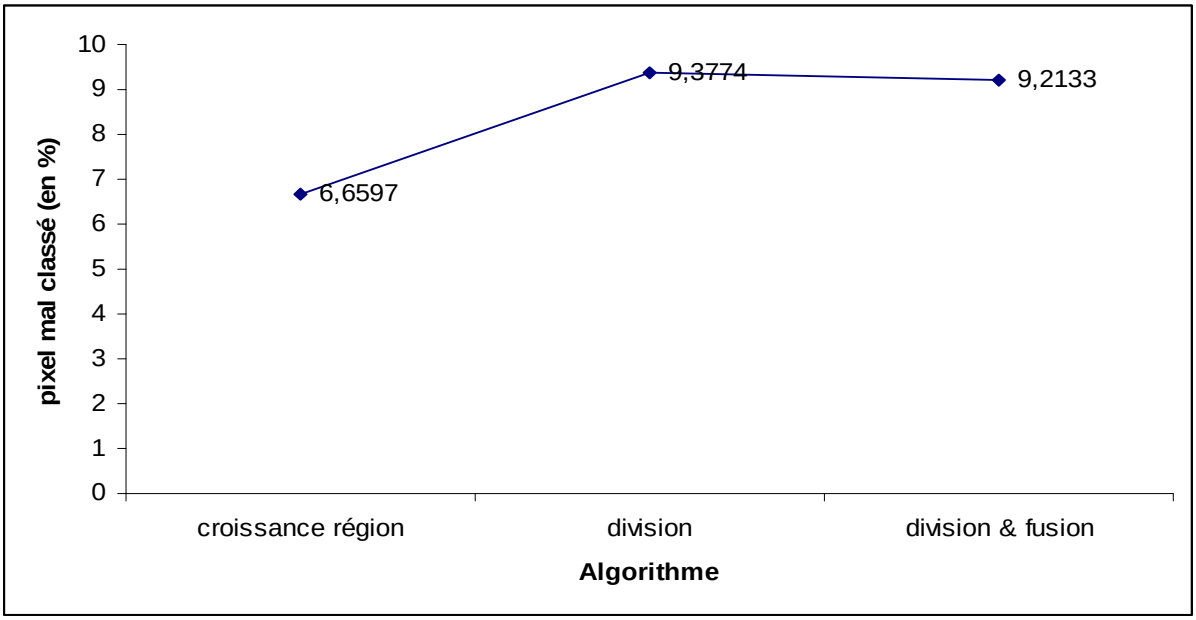


*Figure 2.23* : mesure de performance des algorithmes de segmentation par régions

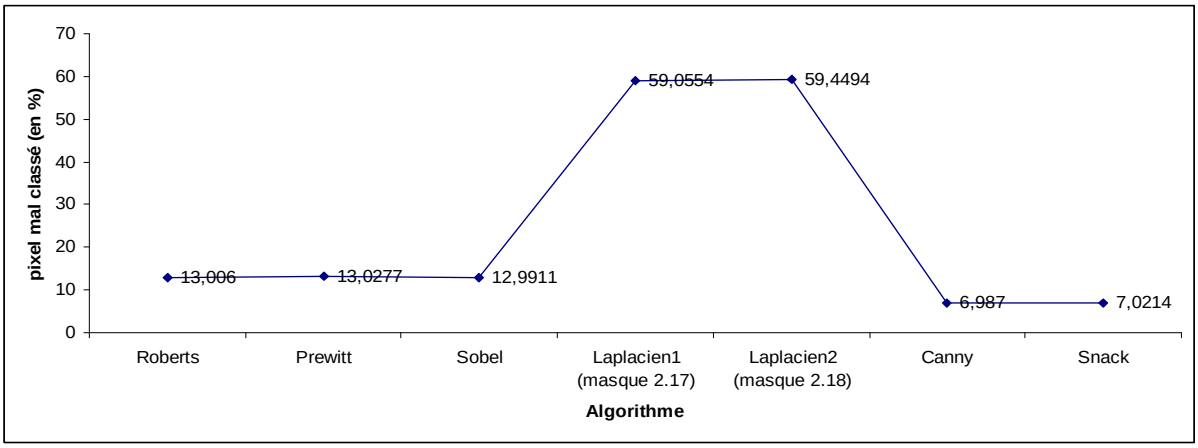


*Figure 2.24* : mesure de performance des algorithmes de segmentation par contours

Le taux de pixel mal classé pour les approches seuillages est un peu élevé, dû à la négligence de la position spatiale des pixels dans l'image.

Avec les approches contours : l'extraction des points contours avec le principe de laplacien n'est pas bonne par rapport au gradient. C'est la raison pour laquelle que Canny a développé son filtre en utilisant le gradient avec un traitement préalable (réduction du bruit) et un post-traitement (seuillage). Le filtre de Canny est donc classé plus performant suivi du contour actif.

Pour les approches régions : le taux est minimal, parce que la technique se base sur les critères de ressemblances entre les pixels voisins. Comme la croissance régions s'étend d'un pixel à une région, le nombre de pixel mal classé est abaissé.

Selon les trois courbes, la segmentation avec l'algorithme de Canny, la croissance région et le seuillage adaptatif sont les plus performantes. Cependant, si l'image à segmenter ne met pas en évidence le fond et les objets présents, le seuillage adaptatif sera éliminé dans le classement.

## 2.7. Domaine d'application de segmentation d'image

### *2.7.1. Télécommunication*

Un exemple d'application de la segmentation d'image en télécommunication est la visioconférence immersive. Afin d'augmenter la vitesse de transmission des données au cours de ce type de visioconférence, la segmentation en temps réel est nécessaire [06].

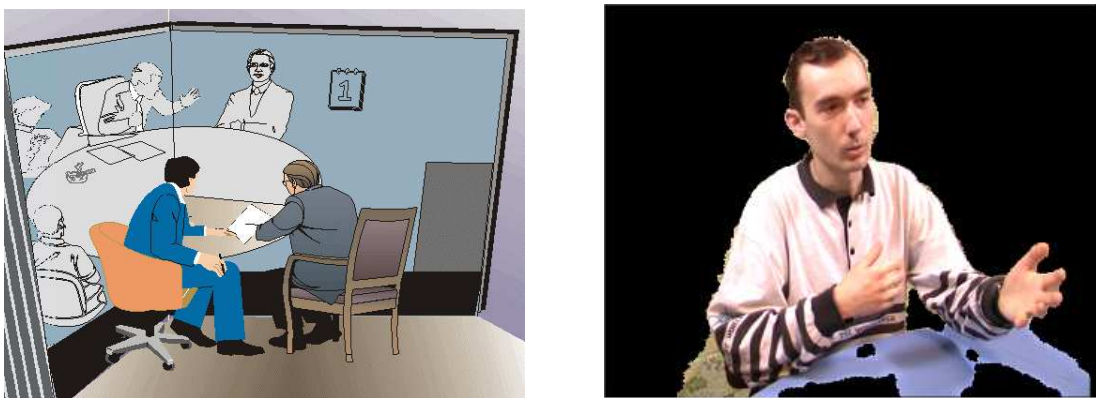

*Figure 2.25* : visioconférence immersive

### *2.7.2. Astronomie*

La segmentation d'image est très utile dans le domaine de l'astronomie en particulier la détection de traces d'astéroïdes. Les figures en dessous vous aident à comprendre bien claire cet utilité.

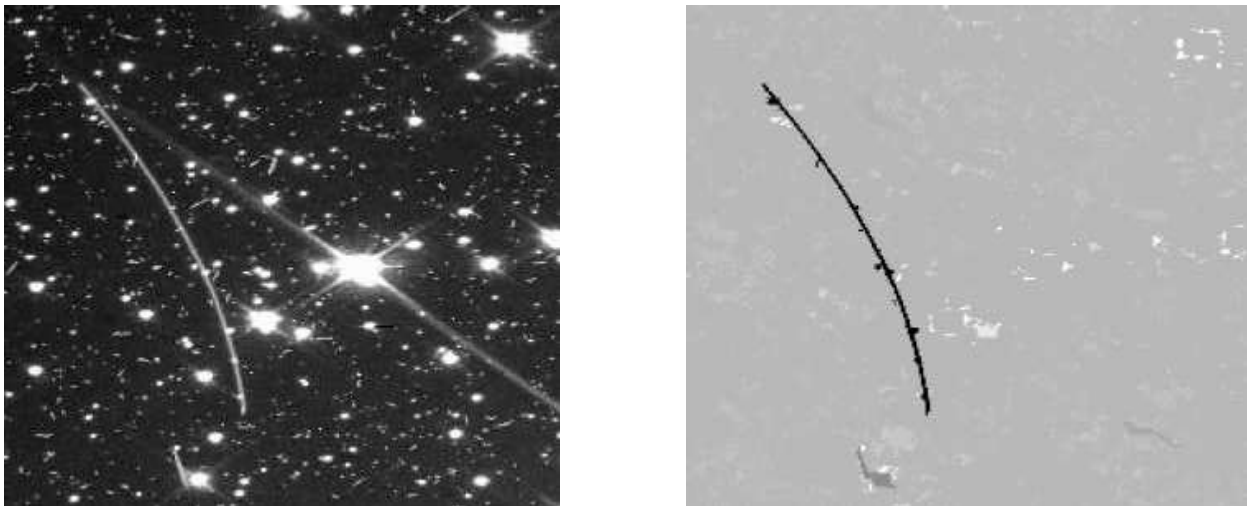

*Figure 2.26* : détection de trace d'astéroïde

### *2.7.3. Recherche des images sur les moteurs de recherches*

De nos jours, le nombre d'images stockées devient de plus en plus grand. Chacun publie ses images sur le Web sans pour autant y ajouter de bons mots clés. Il est maintenant difficile de

réaliser une recherche précise. Les moteurs de recherche actuels se basent sur les mots clés compris dans le texte ou dans le nom du fichier image.
Un meilleur moteur serait vraiment très utile en basant sur l'image elle-même. Par exemple, si on cherche un objet, des images vont apparaître dont le contenu de celles-ci contiennent l'objet en question [21].

### *2.7.4. Application aux images radar*

Nous appliquerons la technique de segmentation pour segmenter une image Radar. On prendra un exemple précis comme le champ de riziculture en Indonésie. L'image ci-dessous montre deux cultures de riz décalées dans le temps. Les zones les plus claires ne sont pas cultivées en riz. Une quatrième classe peut être déterminée pour classer les zones qui n'appartiennent pas aux régions précédentes. Nous donnons dans la suite l'image originale et l'image segmentée [24].

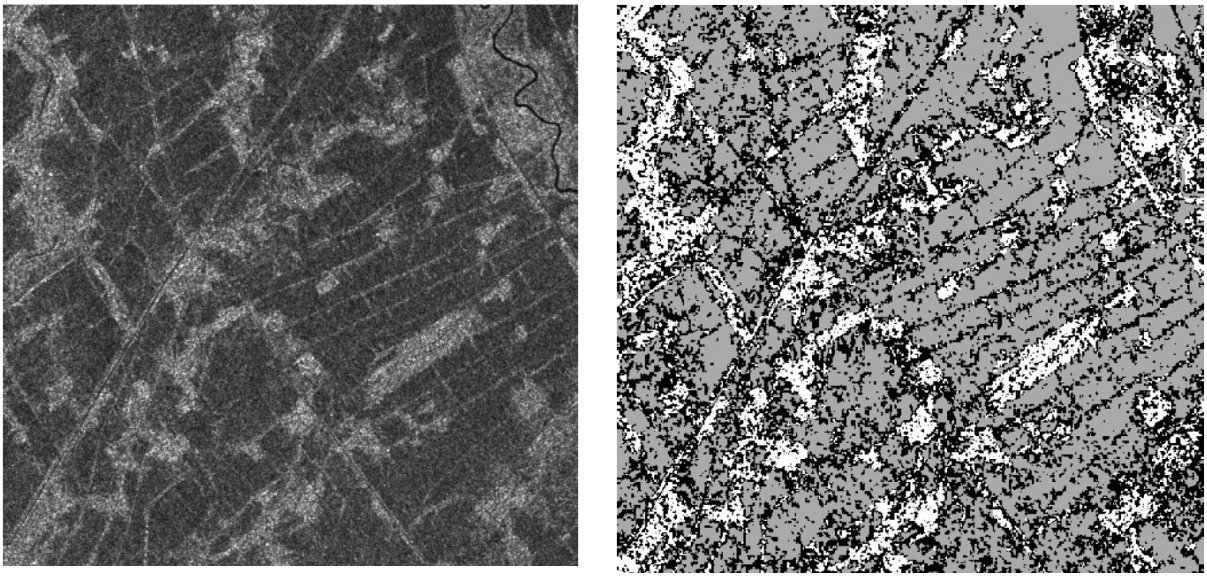

*Figure 2.27* : segmentation image radar

### *2.7.5. Imagerie médicale*

La segmentation d'image en médecine en radiographie par exemple est essentiellement utilisée pour avoir une meilleur qualité et visibilité des organes sur l'image et permettre ainsi de lever des indécisions. Les médecins sont aujourd'hui confronté à l'utilisation d'ordinateurs pour faire un diagnostique plus complet lors de la suspicion d'une anomalie anatomique sur l'image [22].

## 2.8. Conclusion

Dans ce chapitre, nous avons présenté un état de l'art sur les différentes techniques de segmentation d'images. La diversité des techniques existantes nous donne à peu près une idée sur la difficulté du problème. Du fait de cette diversité, il est difficile de définir, de manière absolue, une bonne méthode de segmentation.

Principalement, les méthodes de segmentation sont classées en trois catégories : les méthodes par seuillage, méthodes orientées régions et les méthodes orientées contours.

La méthode par seuillage se focalise sur l'histogramme de l'image à partir duquel on classifie les niveaux de gris sans tenir compte de la position des pixels.

Les méthodes orientées régions quand à elles, se basent sur la similarité dans l'image, et tentent de partager l'image en régions homogènes. Parmi ces méthodes nous trouvons : la segmentation par croissance de régions, fusion de régions, division de régions …etc.

Les méthodes orientées contours se basent sur la dissimilarité dans l'image, ou sur la variation d'intensité entre les pixels, c'est-à-dire là où le changement d'intensité est brusque. Parmi ces méthodes là, nous pouvons citer la segmentation par contour actif qui est la plus utilisée surtout dans le domaine d'imagerie médicale.

Avec ces approches ayant toutes les trois des avantages et des inconvénients, mais grâce à l'étude comparative que nous avons faite au paragraphe 2.6 et des analyses, la segmentation par croissance région possède un taux de pixels mal classés minimum à 6,6597%. Dans le chapitre qui suit, on va essayer d'améliorer cette technique afin de mieux visualiser les objets présents dans une image et de diminuer le pourcentage de pixels mal classés.

# CHAPITRE 3
# AMELIORATION DE SEGMENTATION PAR CROISSANCE REGION

## 3.1. Introduction

### *3.1.1. Description du problème actuel*

Malgré l'existence de nombreuse technique de segmentation d'images, aucune d'entre elles n'est parfaite et la recherche d'une méthode performante est toujours en cours, cela est encore un sujet typique et tous les chercheurs dans le domaine d'imagerie s'intéressent à ce sujet.

Au sujet de contours, le problème persiste encore sur l'apparition des faux contours et de contours non fermés. Jusqu'ici, la technique de contour jugée performante est le filtre de « Canny » suivie de l'algorithme du contour actif connu sous le nom de « snake ».

Avec l'approche région, elle ne fournisse pas exactement la forme des objets présents dans l'image. Si deux objets de même couleur ou avec quelque nuance de niveau de gris se collant entre eux, l'approche région n'arrive pas à détecter la frontière de séparation des deux objets en question après segmentation, les deux objets seront considérés comme un seul objet. Quant à la segmentation par seuillage, elle ne rend pas compte la position spatiale de niveau de gris, en plus si l'histogramme de l'image ne présente pas de vallée, la recherche des seuils devient si trop compliquée.

### *3.1.2. Objectif de la méthode proposée*

L'objectif principal c'est de trouver une méthode permettant d'améliorer la segmentation par croissance région qui est encore classée jusqu'ici parmi la segmentation plus performante. La raison de ce choix a été développée au chapitre précédent.

Deux phases seront utiles pour mener à bien cet objectif :

- Il y aura tout d'abord un *prétraitement* permettant d'améliorer les caractéristiques de l'image à segmenter.
- Il y aura ensuite la *segmentation proprement dite*.

## 3.2. Principe de l'amélioration

### *3.2.1. Prétraitements*

La segmentation par région est basée sur la ressemblance des pixels voisins. Pour que ce principe soit plus efficace, il est nécessaire préalablement de filtrer l'image originale par un filtre médian

afin que les pixels voisins ont presque ou égale la même valeur. Ce type de filtre atténue la variation brusque sur les valeurs de pixels voisins, ce qui permettra de rendre difficile la détection de contour. Pour surmonter à ce problème, une solution efficace est proposée, il s'agit de détecter les contours de l'image originale avec « Canny » puis on procède à éliminer les faux contours, ensuite appliquer à l'image filtrée le résultat du contour. A propos du filtre médian, la dimension de la fenêtre sera fixée à trois ou à cinq selon la dimension de la matrice de l'image à segmenter.

3.2.1.1. Illustrations

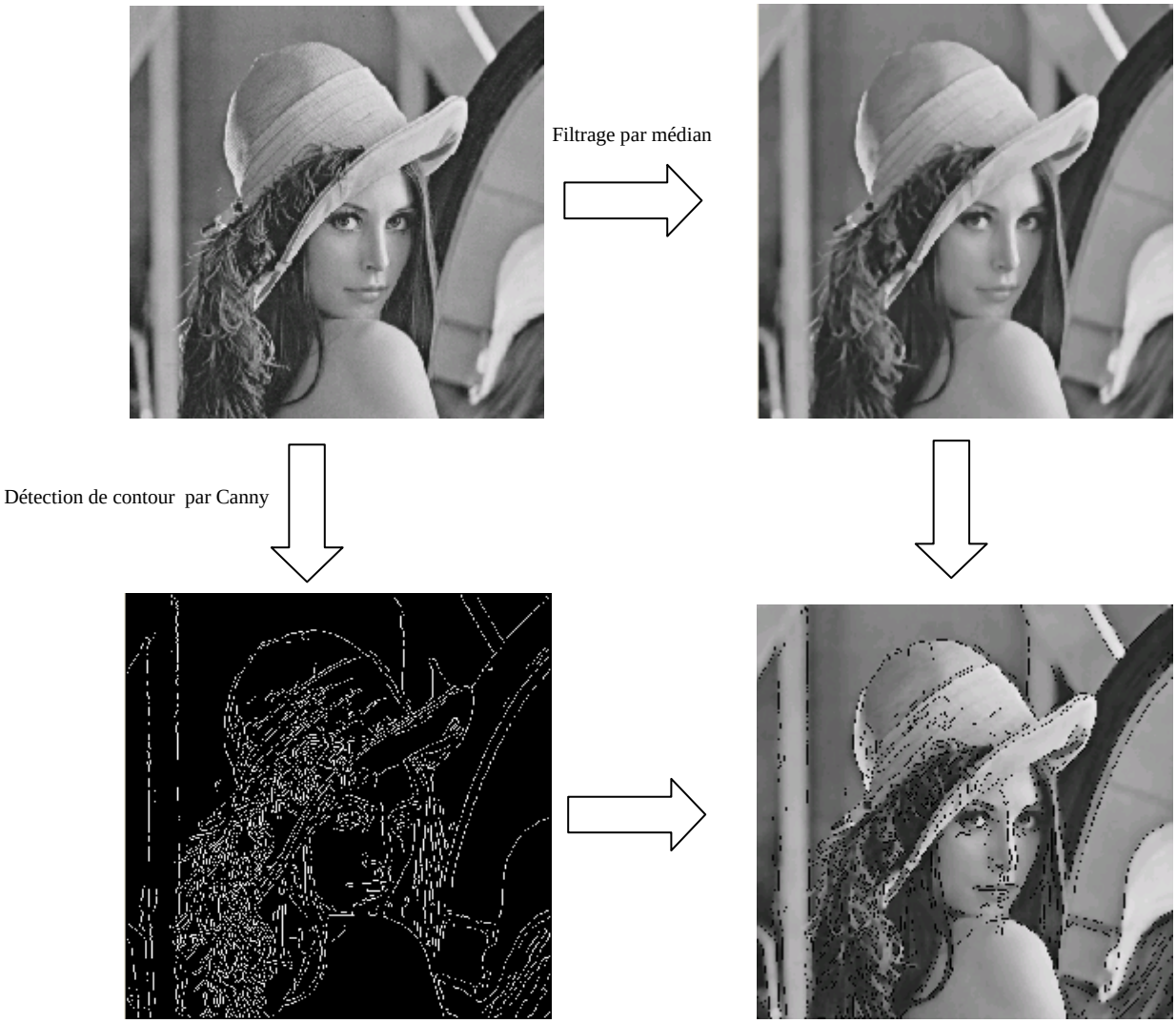


*Figure 3.01* : étapes de prétraitements

### *3.2.2. Segmentation proprement dit : croissance région (récapitulation approfondie)*

Comme les contours sont bien ajustés et les niveaux de gris sont biens lissés, on pourra passer à la segmentation proprement dit qui est la segmentation par croissance région. Dans ce paragraphe, on entrera plus en détail sur la technique de segmentation par croissance région.

#### 3.2.2.1. Présentation de la croissance de régions

Partant d'une région, cette méthode agrège les points situés à la périphérie de cette région respectant une condition d'agrégation. Cette condition d'agrégation d'un point à une région est un critère basé sur une mesure d'homogénéité entre les intensités des points de la région et l'intensité de point à agréger. Chaque agrégation constitue une croissance de la région précédente. Le processus d'agrégation est répété tant que la région évolue.

#### 3.2.2.2. Algorithme de croissance de régions

L'algorithme de croissance de régions consiste, à partir d'un pixel de départ, appelé germe initial, à déterminer itérativement la région en gardant les pixels connexes à la région de l'itération précédente qui satisfont un critère.

Cet algorithme fonctionne donc de manière itérative et se termine lorsqu'il n'y a aucun pixel adjacent à la région satisfaisant le critère. Ce critère n'est pas du tout souple, un pixel est intégré à la région s'il le satisfait complètement. Cet algorithme ne prend pas en charge les pixels isolés qui peuvent contenir des informations pertinentes.

| Algorithme 08: croissance région |
| --- |
| Initialiser le germe initial ou les germes initiaux<br>Agréger les pixels qui respectent la mesure d'homogénéité à la périphérie de chaque germe. |

#### 3.2.2.3. Les paramètres de la croissance de régions

Pour segmenter une image avec la méthode de croissance de régions, il faut faire un certain nombre de choix tels que le positionnement des germes initiaux, la représentation des germes initiaux, la mesure d'homogénéité et le critère d'arrêt.

*3.2.2.3.1. Germes initiaux*

La segmentation par croissance de régions nécessite la détermination des germes définissant la (ou les) première(s) région(s). La position de ce germe influe fortement sur le résultat de la segmentation.

La localisation de ces germes peut être réalisée manuellement. Mais pour certaines applications, cette intervention humaine n'est pas envisageable, il faut alors disposer d'une méthode permettant de localiser automatiquement les germes dans l'image.

Dans la croissance de régions sans germes appelées « *unseeded region growing* », on prend un germe au hasard dans l'image et on construit une première région à partir de ce germe.

Lorsque la région converge, la méthode choisit un autre germe parmi les points non segmentés. La croissance de régions sans germes nécessite fréquemment une étape supplémentaire de fusion de régions pour diminuer le nombre de régions obtenues. La fusion supprime les régions trop petites et regroupe les régions ayant les mêmes caractéristiques.

Un point *p* de l'image ayant une valeur d'intensité est considéré comme un germe si et seulement si un certain nombre de ses voisins ont la même valeur d'intensité que le point *p*.

Le principe de division de régions peut être utilisé pour sélectionner les germes initiaux. La division donne une liste de régions homogènes, il suffit de choisir le centre des plus grosses régions pour avoir de bons germes initiaux.

*3.2.2.3.2. Mesure d'homogénéité*

Pour segmenter une image par croissance de régions, un choix important à faire est la mesure de l'homogénéité. Ce choix est lié à l'application finale et aux propriétés de l'image à traiter. En pratique, cette mesure est souvent une distance entre la valeur de l'intensité ou une autre grandeur propre à l'espace utilisé (niveau de gris, couleur,…) d'un point à agréger et la moyenne et parfois l'écart type des intensités des points constituant la région. Le point est aggloméré à la région en fonction de cette distance. Si cette distance est inférieure à un seuil, ou bien cette distance est la plus petite des distances obtenues avec tout autre pixel candidat à l'agglomération, alors le point est ajouté à la région.

*3.2.2.3.3. Un processus itératif*

Suite au choix des germes initiaux et la mesure d'homogénéité, un processus itératif de déformation ajoute progressivement des points situés à la périphérie de la région en train de

croître, s'ils respectent la mesure d'homogénéité. Ces nouveaux points sont regroupés dans l'ensemble que l'on note $C^{[n]}$ appelé couronne. Considérons une croissance de région ne faisant croître qu'une seule région. Étant donné une région initiale $R^{[0]}$, la région suivante $R^{[n+1]}$ est obtenue à partir de la région courante $R^{[n]}$ et de la couronne $C^{[n]}$ par la réunion suivante :

$$R^{[n+1]} = R^{[n]} \cup C^{[n]} \tag{3.01}$$

*3.2.2.3.4. Convergence, critère d'arrêt*

Comme l'itération porte sur une région croissante et bornée, la convergence est assurée au bout d'un nombre fini d'itérations. Le processus de croissance peut s'arrêter selon deux scénarios :

- toutes les régions satisfont la mesure d'homogénéité et la segmentation comprend n régions
- il existe n-1 régions qui vérifient la mesure d'homogénéité, la $n_{ième}$ région comprend les points ne vérifiant pas la mesure d'homogénéité.

3.2.2.4. Propriétés de la croissance de régions

La croissance de régions a été employée à de multiples reprises pour la segmentation des images médicales. Cela s'explique par le fait que c'est une méthode simple à initialiser et rapide. Dans cette section, les propriétés intrinsèques de cette méthode sont mises en évidence et illustrées sur des données médicales.

Quatre propriétés ont été retenues :

- La croissance de régions permet de segmenter un objet d'intérêt par la simple désignation de cet objet. Cette particularité la rend attrayante pour les applications médicales.
- La croissance de régions ne nécessite pas de lisser le contour de la région segmentée contrairement à d'autres méthodes comme les contours actifs.
- Une propriété consécutive au point précédent est à l'origine de la principale faiblesse de la méthode : sa sensibilité aux points de fuite. Un point de fuite se définit par la connexion de deux régions distinctes aux propriétés statistiques semblables.

- Cette méthode tolère les changements topologiques de la carte de segmentation (région) au cours de son évolution.

### *3.2.3. Algorithme d'amélioration de croissance régions*

Avec cet algorithme, une intervention manuelle n'est pas nécessaire. Les paramètres (recherche germe, critère d'arrêt, … etc.) exigés par la croissance régions sont fournis automatiquement. Les suivantes les étapes à suivre :

*Etape n°1*

*Filtrer l'image originale par un filtre médian.*

Le choix du filtre médian est dû au faite qu'il supprime les bruits impulsionnels, alors il aura beaucoup de ressemblance entre les pixels voisins, c'est-à-dire plus d'homogénéité.

*Etape n°2*

*Détecter les contours de l'image originale avec le filtre de « Canny » puis éliminer les faux contours.*

L'image originale est prise puisque l'image résultante de la première étape n'accentue plus les variations des intensités lumineuses. La recherche des contours sera difficile.

Pourquoi le filtre de Canny ? : D'après les études et analyses au deuxième chapitre, le filtre de Canny possède le taux de pixel mal classé minimum parmi les algorithmes de contours.

*Etape n°3*

*Mettre au dessus de l'image résultante de l'étape n°1, l'image issue de celui de n°2.*

On choisit l'image résultante de l'étape n°1 grâce à la complicité des pixels entres eux.

Le principe de superposition se fait comme suit : quand on détecte un pixel dont sa couleur est blanche sur l'image de l'étape n°2, le niveau de gris correspondant sur l'image filtrée sera mis à zéro.

*Etape n°4*

*Marquer et compter les contours.*

*Etape n°5*

*Trouver le premier germe initial à l'intérieur d'un contour pris au hasard en cherchant le niveau de gris dominant.*

*Etape n°6*

Déterminer les autres germes à l'intérieur des contours à partir du germe initial de l'étape n°5.
Les germes à l'intérieur des contours voisins doivent être différents les uns des autres. S'ils ont le même, on n'arrive pas à distinguer les objets qui se collent entre eux.

*Etape n°7*

Appliquer la croissance région avec paramètres suivants :

- Germes initiaux : résultats de l'étape n°5 + étape n°6
- Critères d'homogénéité :
  - voisinage = 8 (analyse des 8 pixels voisins pour l'agrégation d'un point à une région)
  - seuil : aucune définition de seul
- Critère d'arrêt de l'algorithme : détection de contours

| Algorithme 09: amélioration croissance région |
|---|
| Télécharger l'image à segmentée $I_0$<br>Filtrer $I_0$ par un filtre médian ($I_1$)<br>Détecter le contour de $I_0$ par un filtre de Canny ($I_2$)<br>Eliminer les faux contours ($I_3$)<br>Tant que le niveau de gris dans $I_3$ soit blanc<br>Changer le niveau de gris correspondant sur $I_1$ par noir<br>Fin tant que<br>Choisir un niveau de gris dominant dans un contour fermé comme germe initial.<br>Rechercher les autres germes initiaux en cherchant les contours voisins et choisir un niveau de gris dominant qui est différent du premier germe à l'intérieur du contour en question.<br>Appliquer la croissance région. |

### 3.3. Tests et résultats de l'algorithme

Ici, on se contente seulement sur les aspects visuels des images et les mesures de performances seront entamées au paragraphe suivant. Les images montrées en dessous vont inclure aussi les résultats de croissance régions pour que nous puissions visualiser la différence avec les résultats du nouvel algorithme.

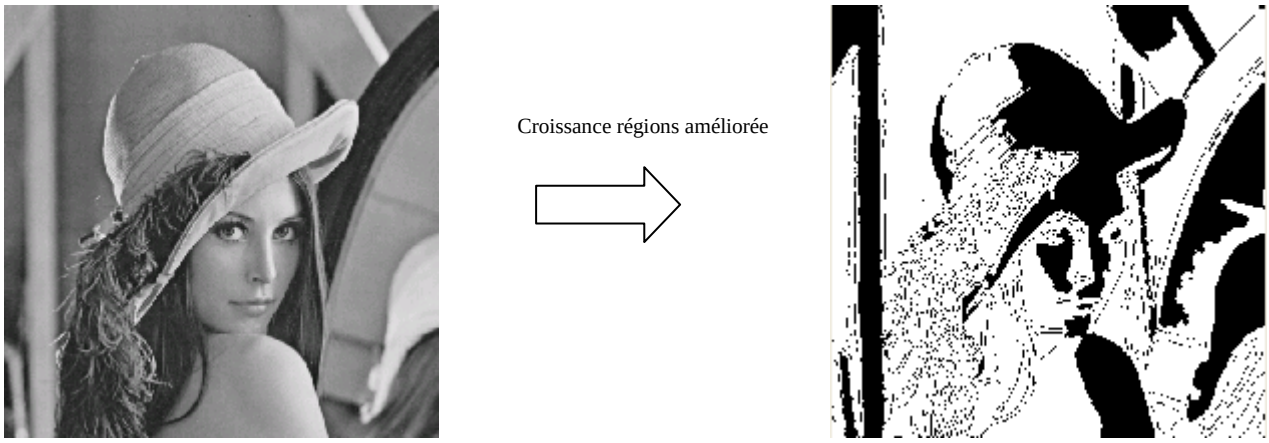


*Figure 3.02* : image originale et segmentée de Lena par l'amélioration de croissance régions

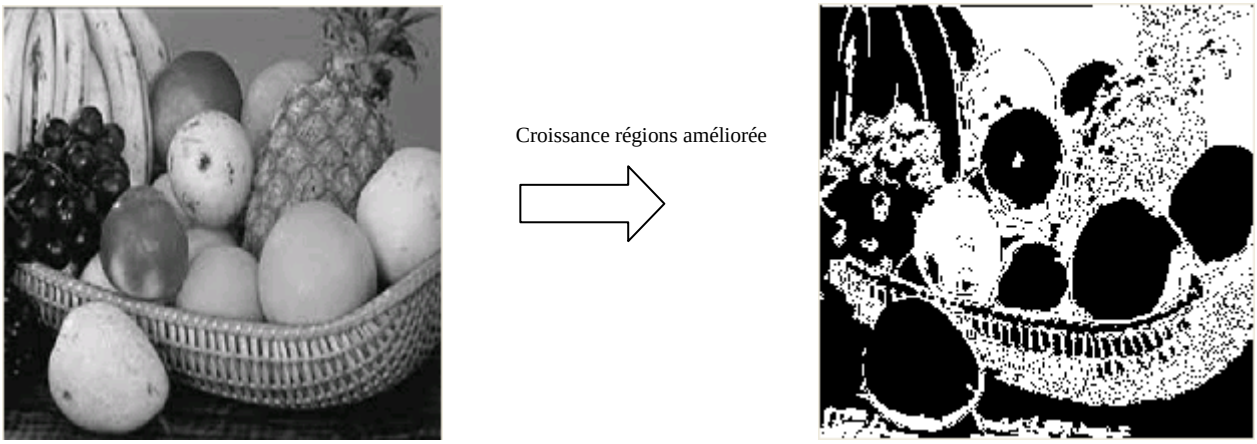


*Figure 3.03* : image originale et segmentée par l'amélioration de croissance régions


*Figure 3.04* : Image de Lena agrandie segmentée par croissance régions


*Figure 3.05* : Image de Lena agrandie segmentée par amélioration croissance régions


*Figure 3.06* : fruits segmentés par croissance régions

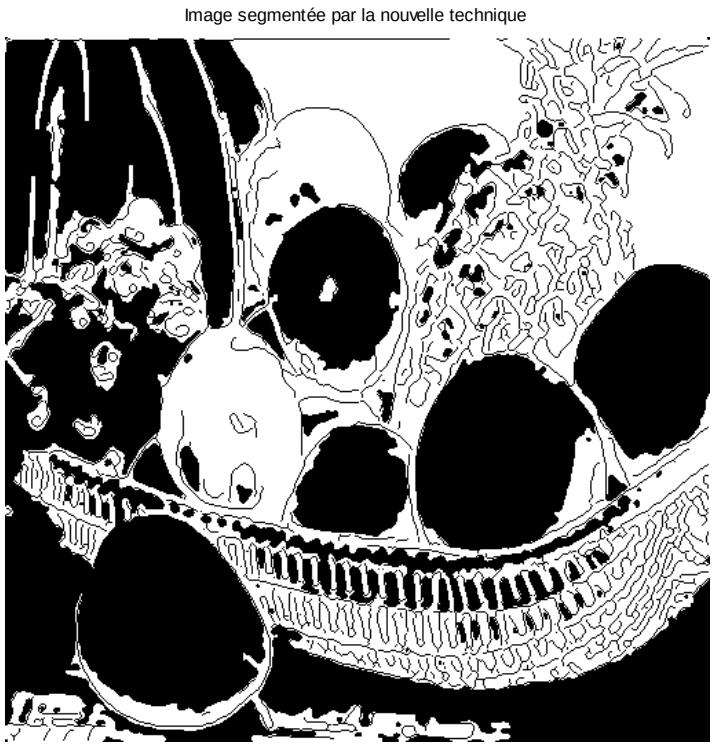


*Figure 3.07* : fruits segmentés par amélioration croissance régions

Après la visualisation, constatation et analyse, nous pouvons citer quelques points forts de l'algorithme :

- La forte densité des informations extraites.
- La meilleure précision de la limite des régions.
- La bonne correspondance entre les régions et les objets contenus dans l'image.
- La non exigence des paramètres ou des critères préalables et la rapidité de la segmentation.

Par contre, son inconvénient est le non performance d'élimination des faux contours même si cela existe dans l'algorithme.

### 3.4. Indicateur de performance

Avec le même principe qu'au paragraphe 2.6.2, nous arrivons à obtenir un taux de pixels mal classés à 4,1222 qui est inférieur à 6,6597%.

Pour calculer le taux de pixel mal classé, nous avons besoin du nombre de pixel mal classé. Ce dernier est égal au nombre des pixels qui ne subissent pas de changement après la segmentation.

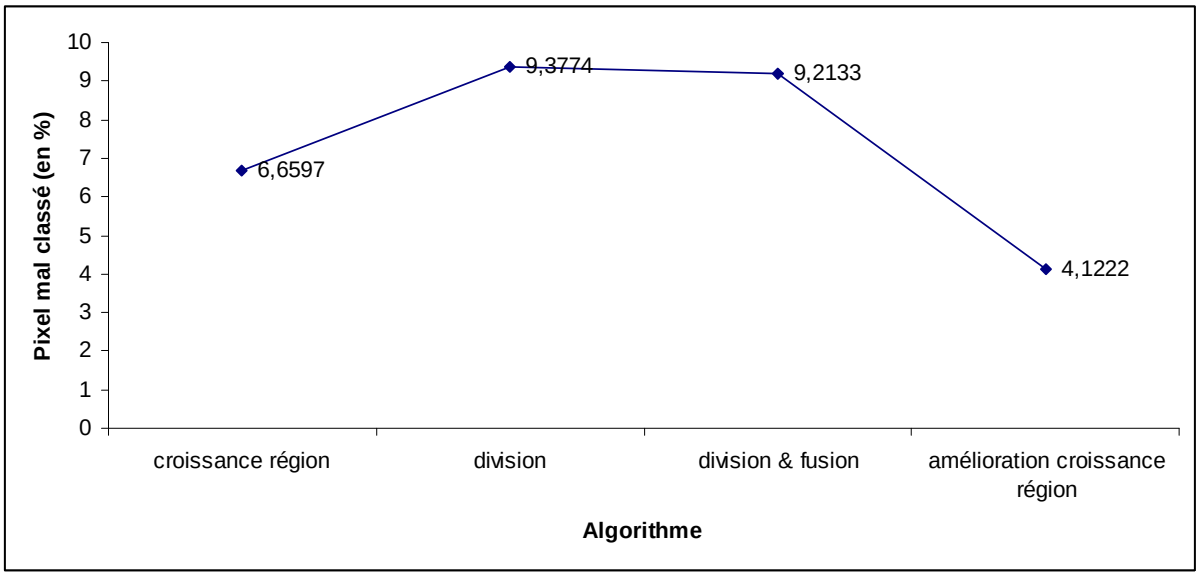


*Figure 3.08*: mesure de performance des algorithmes de segmentation par régions

### 3.5. Conclusion

Chacune des méthodes de segmentation a ses avantages, mais a également ses inconvénients. Comme dans la plupart des problèmes, c'est une histoire de compromis.

Les méthodes demandant des paramètres sont très délicates à manipuler. Elles fonctionnent correctement dans le cas où les paramètres sont choisis judicieusement. Avec la méthode par croissance régions, le mauvais choix des germes initiaux entraine une sous et sur-segmentation de l’image. En plus, la limite des objets ne sont pas claires si le seuil d’agrégation est mal choisi. L'idéal serait de développer des algorithmes permettant de déterminer ces paramètres de façon optimale ce qui était l’objet de ce dernier chapitre en cherchant de façon automatique les germes initiaux et de définir les critères d’homogénéité.

# CONCLUSION GENERALE

Le travail présenté dans ce mémoire a pour objectif d'enrichir la connaissance sur le traitement d'image en se focalisant sur les méthodes de segmentation.
Nous avons entamé notre document par une introduction au domaine du traitement d'images.
Différentes notions relatives au traitement d'image ainsi que les étapes du processus de ce dernier y ont été présentées. Le but était de nous familiariser avec le domaine de l'imagerie
(Chapitre 1). Après le traitement d'image, l'état de l'art de segmentation d'images fait l'objet de second chapitre (Chapitre 2). Nous y avons présenté la segmentation par seuillage en détail en commençant par une présentation de l'approche globale (binarisation manuelle et automatique, seuillage multiple), l'approche locale et l'approche adaptif (algorithme de k-means). Dans ce même chapitre, nous avons abordé aussi la segmentation par régions en développant les méthodes par croissance régions, division, fusion de régions et la combinaison de division et fusion. La segmentation par contours est la dernière méthode entamée dans ce deuxième chapitre à savoir l'approche dérivative (gradient, laplacien, filtre de Canny) et le modèle déformable connu sous le nom du contour actif. Grâce à l'étude des techniques existantes, on arrive à faire une étude comparative et cela nous permet de trouver une idée d'amélioration abordé au chapitre 3. Cette amélioration est déduite à partir des inconvénients avec la croissance région.

Les perspectives de ce travail peuvent être résumées dans les points suivants :

- Etudier et analyser les techniques de segmentations.
- Définir d'autres idées de coopération région-contour, applicable à la technique de segmentation par croissance de région.
- Utiliser les résultats de la segmentation par contour comme solutions initiales pour l'initialisation de germes.
- Définir le critère d'arrêt de l'algorithme de croissance région.

# ANNEXES

## ANNEXE 1 : Algorithme de Canny

L'algorithme de Canny (1986) est utilisé en traitement d'images pour la détection des contours. L'auteur l'a conçu pour être optimal suivant trois critères clairement explicités [11] :

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

### A1.1. Mise en œuvre

#### *A1.1.1. Réduction du bruit*

La première étape est de diminuer le 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, conduisant ainsi à de faux positifs.

Un filtrage gaussien est utilisé, dont voici l'opérateur de convolution :

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad \text{(A.01)}$$

et un exemple de masque 5×5 discret avec σ=1, 4 :

$$h = \frac{1}{189}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} \qquad \text{(A.02)}$$

Habituellement, un filtre est de taille plus réduite que l'image filtrée. Plus le masque est grand, moins le détecteur est sensible au bruit et plus l'erreur de localisation grandit.

### *A1.1.2. Gradient d'intensité*

Après le filtrage, l'étape suivante est d'appliquer un gradient qui retourne l'intensité des contours. L'opérateur utilisé sert à calculer le gradient suivant les directions X et Y, il se compose d'une paire de deux masques de convolution, un de dimension 3×1 et l'autre 1×3 :

$$G_x = \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \quad ; \quad G_y = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} \tag{A.03}$$

La valeur du gradient en un point est approximée par la formule :

$$| G | = | G_x | + | G_y | \tag{A.04}$$

### *A1.1.3. Direction des contours*

Les orientations des contours sont déterminées par la formule :

$$\theta = arctan(\frac{G_y}{G_x}) \tag{A.05}$$

Nous obtenons finalement une carte des gradients d'intensité en chaque point de l'image accompagnée des directions des contours.

### *A1.1.4. Suppression des non-maxima*

La carte des gradients obtenue auparavant apporte une intensité en chaque point de l'image. Une forte intensité indique une forte probabilité de présence d'un contour. Cependant, cette intensité ne suffit pas à décider si un point correspond à un contour ou non. Seuls les points correspondant à des maxima locaux sont reconnus comme correspondant à des contours, et sont conservés pour la prochaine étape de la détection.

Un maximum local est présent sur les extrema du gradient, c'est-à-dire à l'endroit où sa dérivée s'annule.

### *A1.1.5. Seuillage des contours*

La différenciation des contours sur la carte générée se fait par seuillage à hysteresis.

Cela nécessite deux seuils, un haut et un bas; qui seront comparés à l'intensité du gradient de chaque point. Le critère de décision est le suivant. Pour chaque point, si l'intensité de son gradient est :

- Inférieur au seuil bas, le point est rejeté;
- Supérieur au seuil haut, le point est accepté comme formant un contour;
- Entre le seuil bas et le seuil haut, le point est accepté s'il est connecté à un point déjà accepté.

Une fois ceci réalisé, l'image obtenue est binaire avec d'un côté les pixels appartenant aux contours et les autres.

### *A1.1.6. Paramètres*

Les deux paramètres principaux déterminant le temps de calcul et l'acuité de l'algorithme sont la taille du filtre gaussien et les deux seuils.

- Taille du filtre : le filtre utilisé lors de la réduction du bruit a une influence directe sur le comportement de l'algorithme. Un filtre de petite taille produit un effet de flou moins prononcé, ce qui permet la détection de petites lignes bien marquées. Un filtre de taille plus grande produit un effet de flou plus important, ce qui sert à détecter des contours moins nets, par exemple celui d'un arc-en-ciel.
- Seuils : l'utilisation de deux seuils au lieu d'un perfectionne la flexibilité mais certains problèmes propres au seuillage demeurent. Ainsi, un seuil trop bas peut conduire à la détection de faux positifs. Inversement, un seuil trop haut peut empêcher la détection de contours peu marqués mais représentant de l'information utile.

Il n'existe pas aujourd'hui de méthode générique pour déterminer des seuils produisant des résultats satisfaisants sur l'ensemble des types d'images.

**ANNEXE 2 : Code source de segmentation par croissance régions en Java**

Code source Java de segmentation par croissance régions

```
package algorithms.segmentation.regiongrowing;

import javax.media.jai.JAI;
import javax.media.jai.PlanarImage;
import javax.swing.JFrame;
import javax.swing.JProgressBar;
import javax.swing.JScrollPane;
import display.DisplayDEM;
import display.DisplayTwoSynchronizedImages;
import java.awt.Image;

/*
 Ce programme effectue une segmentation d'image par croissance régions avec une interface graphique.
 */

public class DemoSimpleRegionGrowing
  {

 /*
Le point d'entrée de l'application. Nous aurons besoin de donner le nom de fichier de l'image à segmenter.
  */

  public static void main(String[] args)
    {

    if (args.length == 0)
      {
```

```
System.err.println("Usage:javaalgorithms.segmentation.regiongrowing.DemoSimpleRegionGrowi
ng image [preprocess]");
    System.exit(0);
    }

  // Loader l'image.

  PlanarImage image = JAI.create("fileload", args[0]);

  // Créer la tâche de traitement de l'image.

  SimpleRegionGrowing task = new SimpleRegionGrowing(image,(args.length > 1));

  // Créer un affichage de l'image originale et binarisée.

  DisplayTwoSynchronizedImages d =
    new DisplayTwoSynchronizedImages(image,task.getInternalImage());

  // Afficher les images originales et binarisée dans un JFrame.

  JFrame origFrame = new JFrame();
  origFrame.setTitle("Original Image and Binarized Image");
  origFrame.getContentPane().add(new JScrollPane(d));

  // Définir l'opération de fermeture si l'application est terminée.

  origFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
  origFrame.pack(); // ajustement de la taille du frame.
  origFrame.setVisible(true); // afficher le frame.

  // Créer une ProgressBar dans une JFrame.
```

```
JProgressBar progressBar = new JProgressBar(0,(int)task.getSize());
progressBar.setValue(0);
progressBar.setStringPainted(true);
JFrame progressFrame = new JFrame();
progressFrame.setTitle("Progress");
progressFrame.getContentPane().add(progressBar);

// Définir l'opération de fermeture si l'application est terminée.

progressFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
progressFrame.pack();
progressFrame.setVisible(true);

// Nous sommes prêts à faire la segmentation.

task.start();

// Modifier la barre de progression alors que la segmentation est effectuée.

while(!task.isFinished())
  {
  progressBar.setValue((int)task.getPosition());
  progressBar.repaint();
  }

// Segmentation est terminée.

progressBar.setValue((int)task.getPosition());

// Création d'un nouveau frame pour afficher le résultat.

JFrame resultsFrame = new JFrame();
```

```
  resultsFrame.setTitle("Segmentation results");
  resultsFrame.getContentPane().add(new JScrollPane(new DisplayDEM(task.getOutput())));

  // Définir l'opération de fermeture si l'application est terminée.

  resultsFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
  resultsFrame.pack();
  resultsFrame.setVisible(true);

  // Voyons quelques données textuelles sur la segmentation.

  System.out.println("Number of regions: "+task.getNumberOfRegions());
  for(int c=1;c<=task.getNumberOfRegions();c++)
    System.out.println("Region "+c+": "+task.getPixelCount(c)+" pixels");
  }

}
```

**ANNEXE 3 : Code source d'amélioration de segmentation par croissance régions en Matlab**

Algorithme : Code source Matlab d'amélioration par croissance régions

```
function varargout = nouvelle(varargin)

% NOUVELLE M-file for nouvelle.fig
%      NOUVELLE, by itself, creates a new NOUVELLE or raises the existing
%      singleton*.
%
%      H = NOUVELLE returns the handle to a new NOUVELLE or the handle to
%      the existing singleton*.
%
%      NOUVELLE('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in NOUVELLE.M with the given input arguments.
%
%      NOUVELLE('Property','Value',...) creates a new NOUVELLE or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before nouvelle_OpeningFunction gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to nouvelle_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Copyright 2002-2003 The MathWorks, Inc.

% Edit the above text to modify the response to help nouvelle

% Last Modified by GUIDE v2.5 22-Sep-2010 09:40:47
```

```
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;

gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @nouvelle_OpeningFcn, ...
                   'gui_OutputFcn',  @nouvelle_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);

if nargin && ischar(varargin{1})
   gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
   [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
   gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before nouvelle is made visible.
function nouvelle_OpeningFcn(hObject, eventdata, handles, varargin)

% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to nouvelle (see VARARGIN)

% Choose default command line output for nouvelle
handles.output = hObject;
```

```
% Update handles structure
guidata(hObject, handles);

% This sets up the initial plot - only do when we are invisible
% so window can get raised using nouvelle.
%if strcmp(get(hObject,'Visible'),'off')
 %   plot(rand(5));
%end

set(handles.axes1,'Visible','off')
set(handles.axes2,'Visible','off')
set(handles.axes3,'Visible','off')
set(handles.axes4,'Visible','off')
set(handles.axes5,'Visible','off')
set(handles.axes6,'Visible','off')

% UIWAIT makes nouvelle wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = nouvelle_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global I Imfin sary
axes(handles.axes2);
cla;
%I=imread(sary);
%Imfin=nagao(I);
switch sary
   case 'fruit1s.jpg'
        I=rgb2gray(I);
        Imfin=medfilt2(I,[5 5]);
        imshow(Imfin);
   otherwise
        Imfin=medfilt2(I,[5 5]);
        imshow(Imfin);
end

global g g1 can im reg
[can im reg]=posttraitement(Imfin);
%[g, g1]=croissance(Imfin);
axes(handles.axes3);
cla;
imshow(im);
% --------------------------------------------------------------------
function FileMenu_Callback(hObject, eventdata, handles)
% hObject    handle to FileMenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% --------------------------------------------------------------------
function OpenMenuItem_Callback(hObject, eventdata, handles)
% hObject    handle to OpenMenuItem (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)
file = uigetfile('*.fig');
if ~isequal(file, 0)
    open(file);
end

% --------------------------------------------------------------------
function PrintMenuItem_Callback(hObject, eventdata, handles)
% hObject    handle to PrintMenuItem (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
printdlg(handles.figure1)

% --------------------------------------------------------------------
function CloseMenuItem_Callback(hObject, eventdata, handles)
% hObject    handle to CloseMenuItem (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
selection = questdlg(['Close ' get(handles.figure1,'Name') '?'],...
                     ['Close ' get(handles.figure1,'Name') '...'],...
                     'Yes','No','Yes');
if strcmp(selection,'No')
    return;
end

delete(handles.figure1)

% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: contents = get(hObject,'String') returns popupmenu1 contents as cell array
%        contents{get(hObject,'Value')} returns selected item from popupmenu1
axes(handles.axes1);
cla;
global sary I

popup_sel_index = get(handles.popupmenu1, 'Value');
switch popup_sel_index
   case 1
      sary='Lena.tif';
      I=imread(sary);
      imshow(I);
   case 2
      sary='fruit1s.jpg';
      I=imread(sary);
      imshow(I);
   case 3
      sary='Lena.tif';
      I=imread(sary);
      imshow(I);
   case 4
      sary='Lena.tif';
      I=imread(sary);
      imshow(I);
   case 5
      sary='Lena.tif';
      I=imread(sary);
      imshow(I);
end

% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
```

```
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end

set(hObject, 'String', {'Lena', 'Fruits'});

% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%global Imfin g g1 can im reg
%[can im reg]=posttraitement(Imfin);
%[g, g1]=croissance(Imfin);
%axes(handles.axes3);
%cla;
%imshow(im);
global reg performance_region
axes(handles.axes4);
cla;
imwrite(reg,'croissance.tif');
performance_region=evaluation('croissance.tif')
imshow(reg);
%title('Image segmentée par la nouvelle technique');
```

```
% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%global reg
%axes(handles.axes4);
%cla;
%imshow(reg);
%title('Image segmentée par la nouvelle technique');
figure (01)
global  g reg
imshow(reg);
title('Image segmentée par la nouvelle technique');
figure (02)
imshow(g);
title('Image segmentée par croissance région');

% --- Executes on button press in pushbutton6.
function pushbutton6_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --- Executes on selection change in popupmenu2.
function popupmenu2_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns popupmenu2 contents as cell array
%        contents{get(hObject,'Value')} returns selected item from popupmenu2
```

```
global I g g1 B h performance_amelioration
popup_sel_index1 = get(handles.popupmenu2, 'Value');
%axes(handles.axes5);
switch popup_sel_index1
   case 1
         B=qtdecomp(I,.065);
         axes(handles.axes6);
         cla;
         imshow(B);
         title('Image segmentée par split');
         imwrite(B,'split.tif');


   case 2
         axes(handles.axes6);
         cla;
         h=splitmerge(I,2,@predicate);
         imshow(h);
         title('Image segmentée par split & merge');
          imwrite(h,'split_merge.tif');
   case 3

         [g, g1]=croissance(I);
         imwrite(g,'nouvelle.tif');
         performance_amelioration=evaluation('nouvelle.tif')
         axes(handles.axes5);
         cla;
         imshow(g);
          %title('Image segmentée par croissance région');

end
```

```
% --- Executes during object creation, after setting all properties.
function popupmenu2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end

% --- Executes on button press in pushbutton7.
function pushbutton7_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
close
globale

% --- Executes on button press in pushbutton8.
function pushbutton8_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
close
regions

% --- Executes on button press in pushbutton9.
function pushbutton9_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to pushbutton9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
close
contours


% --- Executes on button press in pushbutton10.
function pushbutton10_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
close
```

## BIBLIOGRAPHIE


[01] Ponce J., Fortsyth D., « *Image Segmentation* », Computer Vision Book, 1994.

[02] Chakib K., « *Généralités sur le traitement d'images* », Cours traitement d'images, 1999.

[03] Mansouri A., Mitiche A. and Vazquez C., « *Segmentation d'images par compétition multirégion* », article, INRS-Télécommunications, Université du Québec.

[04] Ganaoui E., Perrot M., « *Segmentation par la méthode des k-means* », EPFL, 2003.

[05] Vandewalle P., Pasini S., Grandgeorge B., « *Image segmentation* », Image and Visual Representation Group, Ecole Polytechnique Fédérale de Lausanne, 2 juin 2003.

[06] Talbot H., « *Segmentation* », ISBS / ESIEE, 1er semestre 2004-2005.

[07] Bouhlel B., Tlapale E., Youssef M., « *Traitement d'image pour les patients atteints de scotomes* », université Nice sophia Antipolis, AU : 2005-2006.

[08] Desolneux A., « *Quelques méthodes de segmentation d'images* », MAP5, Université Paris 5, ANR Mipomodim, 10 novembre 2006.

[09] Aurdal L., « *Image segmentation beyond thresholding (or how to understand Andy Warhols inspiration)* », Norwegian Computing Center, November 13th 2006.

[10] Benyelloul A., « *Approches de segmentation d'images par méthodes biométriques* », Institut National D'informatique, oued-smar Alger, AU : 2006-2007.

[11] Serabi I. et Dong-Chen H.E., « *Les approches de segmentation d'image par coopération régions-contours* », Centre d'applications et de recherches en télédétection (CARTEL), Département de géomantique appliquée, Université de Sherbrooke, 10 avril 2007.

[12] Lelore T., « *Segmentation d'image Application aux documents anciens* », laboratoire des sciences de l'information et des systèmes, Ecole Polytechnique Université de Nantes, mai 2007.

[13] Lecœur J., Barillot C., « *Segmentation d'images cérébrales : Etat de l'art* », Rapport de recherche n°6306 - version 3 - version initiale Juillet 2007 - version révisée.

[14] Olivier B., « *Traitement d'images numériques* », Analyse d'images: 2ème partie, Département Génie Electrique 5GE – TdSi, 2007.

[15] Mohamed L., « *Segmentation d'images par contour actif en appliquant les algorithmes génétiques* », Institut National D'informatique, oued-smar Alger, AU : 2007-2008.

[16] Samir S., « *Environnement de segmentation d'image à base d'une approche biomimétique* », Institut National D'informatique, oued-smar Alger, AU : 2007-2008.

[17] Galland F., Bertaux N. et Réfrégier P.**, «** *Segmentation d'image par minimisation de la complexité stochastique* », Equipe Physique et Traitement d'Images, Institut Fresnel, UMR CNRS 6133, ENSPM, D.U. St Jérôme, 13397 Marseille cedex 20, France : février 2008.

[18] Chambon S., Dumoulin J. and Subirats P, *« Introduction of a wavelet transform based on 2D matched filter in a Markov Random Field for fine structure extraction: Application on road crack detection »*. Conference on Image Processing: Machine Vision Applications II, San José, États-Unis, janvier 2009.

[19] Papin K., « *Analyse comparative d'algorithmes de Traitements d'images* », Ecole Nationale Supérieure d'Electricité et de Mécanique de Nantes, 2009.

[20] Hadjira B., Melzi S., « *La segmentation d'image par Croissance de régions* », Ecole nationale supérieure d'informatique, AU 2009 / 2010.

[21] Armel S., « *Region Segmentation* », Institut National D'informatique, oued-smar Alger, AU : 2009-2010.

[22] Saada R., Benchabane Y.**,** *« Segmentation d'images médicales volumétriques par champs de Markov cachés »*, Institut National D'informatique, oued-smar Alger, AU : 2009-2010.

[23] Thierry S., « *Segmentation* », sys- 844, Module 4, 2009.

[24] Emani S., « *Image segmentation* », article publié le 12 janvier 2010 à Lausanne.

[25] Denizot Y., « *Traitement d'images médicales* », Biophysique médicale Médecine Nucléaire CHU Angers, mai 2010.

**RENSEIGNEMENTS**

Nom : RAFIDISON
Prénoms : Maminiaina Alphonse
Tél. : +261 34 00 163 16
E-mail : mamynyaina@gmail.com

Titre du mémoire : **CONTRIBUTION A L'AMELIORATION DE LA SEGMENTATION D'IMAGES PAR CROISSANCE REGIONS**

Nombre de pages : 91
Nombre de tableaux : 03
Nombre de figures : 45

Mots clés : image, traitement, segmentation, régions, contours, seuillages, histogramme, germe, homogénéité, voisinage, niveaux de gris, filtre, performance.

Directeurs de mémoire : Monsieur RANDIMBINDRAINIBE Falimanana
Tél. : +261 34 06 466 90

Monsieur RAZAFINDRADIANA Henri Bruno
Tél. : +261 32 02 174 56
E-mail : hbrazafindradina@gmail.com

**RESUME**

La segmentation est une technique qui nous permet d'extraire les objets présents dans une image en la découpant en plusieurs régions ou contours. Elle est très utile dans le domaine médicale, informatique, télécommunication, … etc. Plusieurs algorithmes sont disponibles et on peut les classifier en trois catégories : seuillages, régions et contours. Chaque algorithme a ses avantages et ses inconvénients et en faisant l'étude de performance, le plus efficace c'est la croissance région classifiée parmi l'approche régions.

Par contre, cette dernière n'arrive pas à bien identifier la forme des objets de l'image et en plus, la recherche des germes initiaux compliquent le commencement de la segmentation et la condition d'arrêt de l'algorithme est aussi un facteur bloquant. Afin de remédier à ses problèmes, un nouvel algorithme d'amélioration de la croissance région est mis en place. L'idée est de trouver automatiquement des germes initiaux et la condition d'arrêt de l'algorithme. Le premier germe initial est le niveau de gris dominant dans un contour fermé et à partir de ce dernier on détermine les autres germes en choisissant un pixel dont le niveau de gris ne ressemble pas au germe initiaux des contours voisins et la condition d'arrêt est la détection de contours. Avec ce principe, le pourcentage de pixels mal classés devient minimal et on détecte bien les objets présents dans l'image.

**ABSTRACT**

Image segmentation is the process of partitioning the digital image into multiple regions or edge. The goal of image segmentation is to find regions that represent objects or meaningful parts of objects. It is used for several applications in the field of medicine, in informatics and telecommunication …. We can classify segmentation algorithms into three classes: Threshold, region, and edge. Each algorithm has his own inconvenient and advantage but after performance analysis, the growing region which is part of region segmentation has more efficient.

With this technical, we have a difficulty to find some parameters as initial seed and the condition in which time the algorithm should be stopped. Growing region cannot detect exactly the form of object present on image. So, to resolve these problems, we create a new algorithm to improve growing region. This algorithm searches automatically all initial seed: the first initial seed is the gray value prevailing in a closed edge and from this, we define the other seeds by choosing a pixel whose the gray level does not resemble with the initial seed of edges neighbors. The algorithm will be stopped, when it detects edge. With this technical, the efficiency rate is better than other algorithms and the majority of objects on image are detected.