Université Abou Bekr Belkaid
جامعة ابي بكر بلقايد
تلمسان الجزائر

République Algérienne Démocratique et Populaire
Université Abou Bakr Belkaid– Tlemcen
Faculté de Technologie
Département d'Informatique

Mémoire de fin d'études

Pour l'obtention du diplôme d'Ingénieur d'État en Informatique

Option : Système d'information

Thème

Segmentation des images texturées par la méthode markovienne

Réalisé par :

- **M^elle AZMANI Esma.**
- **M^me BENYELLES Kamila.**

Présenté le 04 juillet 2011 devant les jurys composé de :

Mme: CHAOUCHE. L	*Président*	*U.A.B. Tlemcen*
Mr ::BENAISSA. M	*Encadreur*	*U.A.B. Tlemcen*
Mr: BELABED. A	*Examinateur*	*U.A.B. Tlemcen*
Mr BENMOUNA. M	*Examinateur*	*U.A.B. Tlemcen*

Année universitaire 2010-2011

Table des matières

CHAPITRE 2 : Segmentation des images

CHAPITRE 3 : Analyse des textures

Liste des figures

CHAPITRE 1 : Introduction au Traitement d'Image

CHAPITRE 2 : Segmentation des images.

CHAPITRE 3 : Analyse des textures

CHAPITRE 4 : Segmentation d'image par champ Markov

CHAPITRE 5 : Conception

CHAPITRE 6 : Implémentation et résultats obtenus

Liste des tableaux

Introduction Générale

Introduction Générale

A titre indicatif notons que prés de 90% de l'information reçue par l'homme est visuelle. La production d'images de qualité, de même que leur traitement numérique (et si possible) automatique a donc une importance considérable.

Les premières applications des images numériques remontent aux années 1920 et concernent que les premiers pas en transfert d'images et en correction amélioration d'images.

Le traitement d'image représente des scènes naturelles, introduit un volume très important d'information et exige une élaboration substantielle à tous les niveaux :

1. Prétraitement
2. Segmentation
3. reconnaissance des formes et interprétations.

La prise en compte de toutes ces phases influe indéniablement sur la qualité des taches des traitements et sur le temps d'exécution. Un système de vision doit être doté d'une représentation interne permettant de surmonter ces difficultés. Nous penchons ici sur le problème de l'extraction des indices visuels issus des images texturées dans la phase de la segmentation.

Théoriquement, l'analyse d'images est un ensemble d'approches, de méthodes, de techniques offertes par les mathématiques et le traitement de signal pour pouvoir extraire et comprendre de façon automatique les informations présentes dans une image.
Ces informations peuvent être significatives pour le système visuel ou pour un domaine d'application particulier.

En pratique, l'analyse d'image est une suite de phases qui doivent être exécutées, depuis la formation de l'image jusqu'à la prise de décision fondée sur son contenu.
Certaines de ces phases successives sont souvent étroitement liées et souvent indissociables. les unes sont obligatoires alors que l'exécution de certaines d'autres s'avère parfois facultative.

La segmentation d'image a pour but d'extraire de l'image un certain nombre d'entités appelées objets. Ces entités doivent être représentatives du contenu de la scène pour permettre des traitements optimaux d'extraction de caractéristiques, de reconnaissance, de forme et d'interprétation.

La segmentation est une tâche facile pour notre système visuel. En revanche, elle pose de nombreux problèmes non résolus en analyse automatique. C'est la raison pour laquelle, de nombreux chercheurs s'attèlent à trouver des techniques qui allient, efficacité et temps de calcul réduit. Ainsi, beaucoup de nouvelles méthodes ont été conçues ces dernières années, chacune d'entre elles ayant bien entendu, des avantages et des inconvénients.

Notre mémoire est structuré comme suite :

Chapitre 1 : Le premier chapitre traite sur les notions de bases et des généralités sur l'imagerie et sur les systèmes de traitements d'images dont nous parlons des différentes caractéristiques d'images numériques et les méthodes de prétraitement.

Chapitre 2 : Le deuxième chapitre concerne les grandes familles de segmentation, présentant brièvement le principe de chaque approche.

Chapitre 3 : Le troisième chapitre est consacré à l'analyse de la texture d'image (image texturée).

Chapitre 4 : Le quatrième chapitre concerne notre but qui est la segmentation des images texturées par la méthode markovienne.

Chapitre 5 : Le cinquième chapitre est consacré à l'architecture de notre logiciel ainsi que les différents résultats obtenus lors de son expérimentation sur différentes images texturée

Chapitre 1

Introduction au Traitement d'Image

1.1. Introduction [1]

Le système de vision de l'être humain est un processus compliqué, puisque toute excitation donne une perception colorée, même s'il n'y a pas eu de sensation colorée issue de l'œil, c'est le cas des rêves, par exemple.

La vision intervienne dans un grand nombre d'activités humaines, pour cette raison les chercheurs sont intéressés par l'idée de remplacer l'observateur humain par la machine, cette idée à donner naissance au traitement d'image qui a un champ d'application très vaste.

Au sein de développements forts de la science, on ne peut pas nier le rôle de l'image numérique, un support important des applications dans de très nombreux domaines tels que la médecine, le multimédia, la robotique... Parmi une série d'opérations effectuées sur les images, le traitement d'images est considéré comme une étape de base

Le traitement numérique des images n'est pas un nouveau phénomène. Des techniques pour la manipulation, la correction et le rehaussement d'images numériques sont utilisés depuis plus de trente ans. Nous vivons désormais à l'ère de l'image, du traitement numérique du signal et des télécommunications. Mais sans traitement numérique approprié, une grande partie des images reproduites ou retransmises seraient de piètre qualité.

1.2. La vision [2]

1.2.1. La vision humaine

Une personne regardant autour d'elle peut décrire pratiquement immédiatement ce qu'elle voit.

La scène est perçue à l'aide des cellules de la rétine qui correspondent grosso modo à 1.000.000 pixels (1 pixel = 1 point image), chaque pixel étant quantifié en couleur (rouge, vert, bleu) et en intensité.

Fonctionnement de la perception[S1]

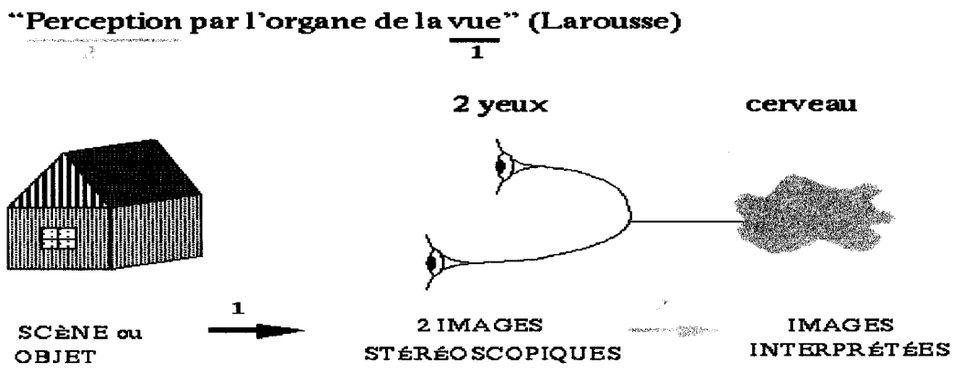


Figure 1.1 : vision humaine

La vision = vue + perception.

1.2.2. La vision par ordinateur

Extraction d'informations à partir d'images représentant un monde tridimensionnel (3D) et de séquences d'images représentant un monde à quatre dimensions (3D + temps).
En final c'est l'ordinateur qui réalise le travail de perception du système de vision humaine de façon automatique.

1.3. Concepts et propriétés d'image [4]

1.3.1. Définition dune Image

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

1.3.2. Définition de l'Image numérique

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 prélevées à l'emplacement correspondant dans l'image réelle.

La numérisation d'une image est la conversion de celle-ci de son état analogique en une image numérique représentée par une matrice bidimensionnelle de valeurs numériques f(x, y) où :

X, Y : Coordonnées cartésiennes d'un point de l'image.

f(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 (2m niveaux de gris ; 0 = noir ; 2m-1 = blanc). La valeur en chaque point exprime la mesure d'intensité lumineuse perçue par le capteur.
La numérisation consiste au passage du monde réel (infini) au monde discret (fini),

1.3.3. Caractéristiques d'une image numérique[2]

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

❖ **Pixel**

Un pixel est l'unité de base d'une image numérique. Il constitue l'unité minimale adressable par le contrôleur vidéo. A chaque pixel est associe une couleur, elle-même décomposée en 3 composantes primaires qui sont le Rouge, le Vert et le
Bleu (RGB).
Les pixels ont une forme rectangulaire proche du carré comme le montre le schéma suivant :

Figure 1.2 : pixel

La quantité d'information que véhicule chacun des pixels donne des nuances entre images monochromes et images couleur. Dans le cas d'une image monochrome, chaque pixel est codé sur un octet, et la taille mémoire nécessaire pour afficher une telle image est directement liée à la taille de l'image.

Dans une image couleur (R.V.B.), un pixel peut être représenté sur trois octets : un octet pour chacune des couleurs : Rouge (R), Vert (V) et Bleu (B).

Par exemple, si pour chaque couleur on a 256 niveaux (2 puissance 8 niveaux (bit)), le point constitué de trois couleurs aura 256x256x256 = 16 777 216 couleurs possibles. On dit que la couleur est codée sur 24 bits (3 fois 8 bits) ou à une profondeur de couleur de 24 bits.

❖ **Dimension**

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

❖ **Résolution**

C'est la clarté ou la finesse de détails atteinte par un moniteur ou une imprimante dans la production d'images. Sur les moniteurs d'ordinateurs, la résolution est exprimée en nombre de pixels par unité de mesure (pouce ou centimètre). On utilise aussi le mot résolution pour

désigner le nombre total de pixels affichables horizontalement ou verticalement sur un moniteur.

❖ **Bruit (parasite)**

Un bruit dans une image est considéré comme un phénomène de brusque variation de l'intensité d'un pixel par apport à ses voisins, il provient de l'éclairage des dispositifs optiques ou d'une baisse momentanée de l'intensité électronique de capteur.

Figure 1.3 : Image bruitée.

❖ **Adjacence**

Un point du maillage sera appelé pixel. Comme le maillage forme un réseau, les pixels sont les points à coordonnées entières selon deux axes. Donc tout pixel se code comme un couple (i, j) d'entiers ; ainsi les coordonnés correspondent à la notation matricielle, le pixel (i,j) se trouvant à l'intersection de la ligne i et de la colonne j.

Dans un maillage carré, tout pixel a 2 types de voisins, à savoir ses 4 voisins selon les axes, et ses 4 voisins selon les diagonales. La relation de proximité entre deux voisins axiaux est plus forte que celle entre deux voisins diagonaux. Par conséquent nous définissons deux relations d'adjacence sur les pixels de ce maillage : deux pixels p et q sont dits :

4- adjacents s'ils sont voisins suivant un axe,

8- adjacents s'ils sont voisins suivant un axe et une diagonale.

Pour le maillage triangulaire (correspondant au pavage hexagonal), il n'y a qu'une seule relation d'adjacence, appelée 6-adjacence. Les 3 relations d'adjacence sont illustrées ci-dessous :

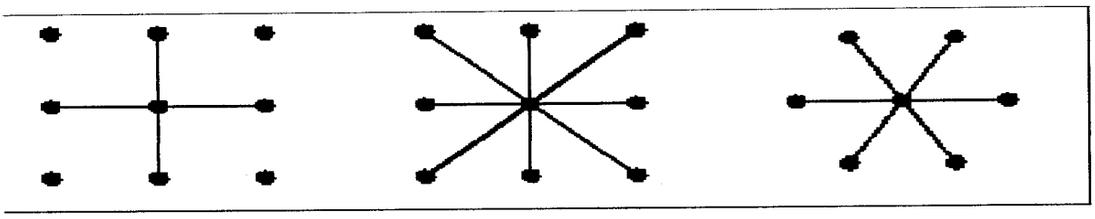


Figure 1.4 : Adjacence de pixels.

❖ **Luminance**

C'est le degré de luminosité des points de l'image. Une bonne luminance se caractérise par :

✓ Des images lumineuses (brillantes);

✓ Un bon contraste : il faut éviter les images où la gamme de contraste tend vers le blanc ou le noir; ces images entraînent des pertes de détails dans les zones sombres ou lumineuses.

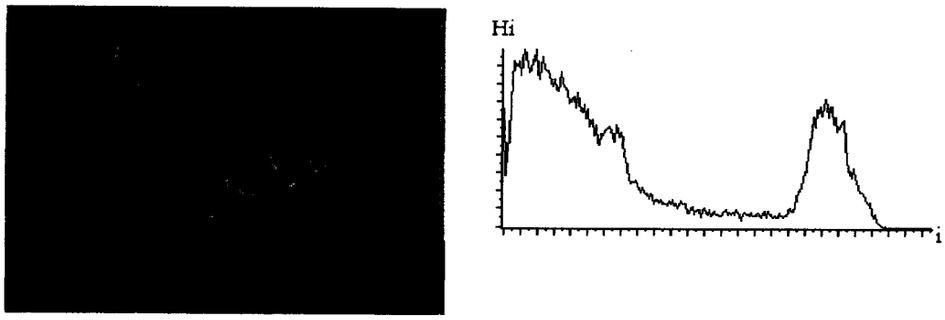
✓ L'absence de parasites.

❖ **Histogramme d'une image**

Un histogramme est un graphique statistique permettant de représenter la distribution des intensités des pixels d'une image. Par convention un histogramme représente le niveau d'intensité en abscisse en allant du plus foncé (à gauche) au plus clair (à droite).

Ainsi, l'histogramme d'une image en 256 niveaux de gris sera représenté par un graphique possédant 256 valeurs en abscisses, et le nombre de pixels de l'image en ordonnées. Prenons par exemple l'image suivante composée de niveaux de gris

L'histogramme et la palette associés à cette image sont respectivement les suivants :



i → H(i) i=0....255

H(i) : Nombre d'occurrences du niveau de gris i dans l'image

Figure 1.5 : Représentation d'un histogramme d'une image

L'histogramme fait apparaître que les tons gris clair sont beaucoup plus présents dans l'image que les tons foncés.

Le ton de gris le plus utilisé est le 11ème en partant de la gauche.

Pour les images en couleur, plusieurs histogrammes sont nécessaires.

- Un histogramme représentant la distribution de la luminance,
- Trois histogrammes représentant respectivement la distribution des valeurs respectives des composantes rouges, bleues et vertes.

Pourquoi utilisons l'histogramme

- Comparer deux images obtenues sous des éclairages différents.
- Mesurer certaines propriétés sur une image.
- 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.
- Améliorer la qualité d'une image (Rehaussement d'image) en introduisant quelques modifications, pour pouvoir extraire les informations utiles de celle-ci.

Parfois dans une image naturelle, une majorité de pixels ont une valeur inférieure à la luminance moyenne. Ceci entraîne un problème qui est les détails dans régions sombres sont difficilement perceptibles ; l'une des techniques utilisées pour pallier cet inconvénient est connue sous le nom « égalisation d'histogramme » pour rendre l'histogramme proche d'une certaine forme bien définie.

❖ **Contours et textures**

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.

❖ **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.

1.4. Différents types d'images [4]

1.4.1. Images Noir et blanc

Ces images sont dites à niveaux de gris, car on ne prend pas en compte ici la couleur mais seulement l'intensité lumineuse.

Parmi ces images on peut trouver:

❖ **Images binaires** : Où chaque pixel est représenté par un bit (0/1) avec en général (0 ---> noir, intensité nulle et 1 ---> blanc, intensité maximale).

❖ **Images en niveaux 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 chacun des pixels de l'image une valeur correspondant à la quantité de lumière renvoyée.

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.

1.4.2. Images couleur

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.).

La représentation en couleurs réelles

Elle consiste à utiliser 24 bits pour chaque point de l'image. Chaque pixel peut prendre une valeur dans le RVB comprise entre 0 et 255 Cela est cependant théorique, car aucun écran n'est capable d'afficher 16 millions de points. Dans le plus haute résolution (1600 x 1200), l'écran n'affiche que 1920000 points. Par ailleurs, l'œil humain n'est pas capable de distinguer autant de couleurs.

Valeur R	Valeur V	Valeur B	Couleur correspondante	Commentaires
0	0	0		noir
0	0	1		un peu moins noir (nuance impossible à détecter à l'œil par rapport au noir)
...	...	...	...	...
0	0	255		Bleu
...	...	...	...	...
0	255	0		Vert
...	...	...	...	...
255	0	0		rouge
...	...	...	...	.
128	128	128		couleur intermédiaire correspondant à un gris
255	255	255		blanc

Tableau1.1 : modèle RVB.

La représentation en couleurs indexées

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.

Palette avec les codes RVB	**Couleurs correspondantes:**
Couleur 0:255 255 255	
Couleur 1:255 255 204	
Couleur 2:255 255 153	....
Couleur 3:255 255 102	
Couleur 4:255 255 51	
Couleur 5:255 255 0	

Tableau 1.2 : Exemple d'une palette de couleur

1.4.3. Images 3D

Des images 3D sont des images qui représentent une scène en trois dimensions. Le « pixel » est alors appelé un voxel, et représente un volume élémentaire.
Ces images peuvent être évidemment, d'un des deux types définis précédemment (N/B ou couleur).
Des exemples d'images de ce type se rencontrent dans les images médicales.

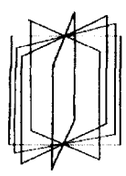


Figure 1.6 :Representation d'une image 3D

1.4.4. Qualité de l'image numérique

Elle dépend, d'une part, de la qualité des images d'origine et, d'autre part, des moyens mis en œuvre pour convertir un signal analogique en signal numérique. Elle dépend aussi de :

- La qualité des périphériques de numérisation de l'image,
- La qualité de l'affichage à l'écran

Nécessité de traiter les images numériques

L'acquisition d'images numériques transforme un signal continu analogique en échantillonnage numérique discontinu. Il y a donc des pertes d'information dues aux méthodes d'échantillonnage et d'autres dues aux erreurs d'acquisition (capteur défectueux, couverture nuageuse, défauts d'optique, etc.).

Le premier problème est résolu par : la fréquence d'acquisition doit être double de la fréquence maximale du signal.

Le deuxième problème est résolu par : une manipulation des données enregistrées. D'où les traitements d'images et les filtrages.

1.4.5. Images Bitmap et image vectorielles

Les images appartiennent à deux grandes familles : bitmap (image-bit) et vectorielle :

- **L'image Bitmap** (Appelées aussi images matricielle): C'est une image numérique représentée par une matrice de pixels ou chacun de ces points possédant une ou plusieurs valeurs décrivant sa couleur.

Avantages d'une image bitmap

1. Adaptable aux images complexes.
2. Adaptable aux traitements d'images

Inconvénients

1. transformation par réduction => perte d'information.
2. Transformation par agrandissement => grossissement des points=> Encombrement important.

➢ **L'image vectorielle** Les images vectorielles sont des représentations d'entités géométriques telles qu'un cercle, un rectangle ou un segment. Celles-ci sont représentées par des formules mathématiques (un carré est défini par deux points, un cercle par un centre et un rayon, une courbe par plusieurs points et une équation). C'est le processeur de l'ordinateur qui sera chargé de «traduire» ces formes en informations interprétables par la carte graphique.

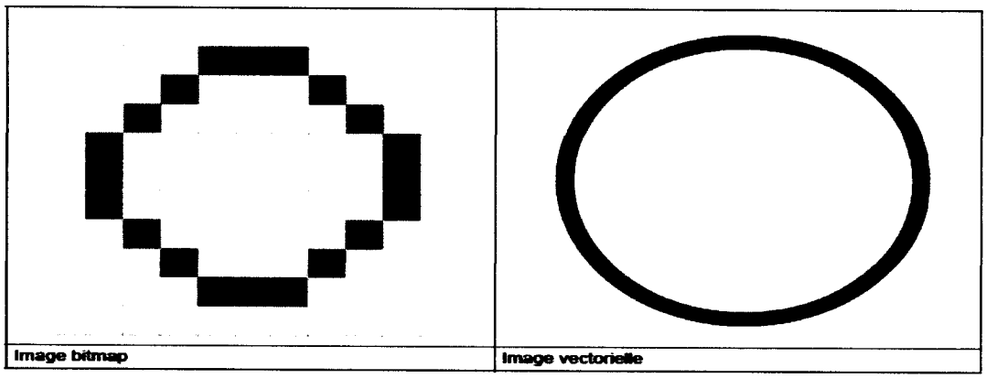


Figure 1.7 : Image vectorielle

1.5. Traitement d'image [2]

Le traitement d'image représente des scènes naturelles, introduit un volume très important d'information et exige une élaboration substantielle à tous les niveaux :

1. Prétraitement
2. Segmentation
3. reconnaissance des formes et interprétations.

Théoriquement, l'analyse d'images est un ensemble d'approches, de méthodes, de techniques offertes par les mathématiques et le traitement de signal pour pouvoir extraire et comprendre de façon automatique les informations présentes dans une image.
En pratique, l'analyse d'image est une suite de phases qui doivent être exécutées, depuis la formation de l'image jusqu'à la prise de décision fondée sur son contenu.
Certaines de ces phases successives sont souvent étroitement liées et souvent indissociables. les unes sont obligatoires alors que l'exécution de certaines d'autres s'avère parfois facultative.

1.5.1. Prétraitement d'images

On regroupe souvent sous le terme de prétraitement toutes les opérations qui sont appliquées aux images, pour leur assurer une bonne qualité. Elles concernent donc essentiellement les corrections de contraste et la suppression du bruit.
L'étape de prétraitement a pour but de faciliter la segmentation en renforçant la ressemblance entre pixels appartenant à une région.

Les prétraitements les plus utilisés :

- La modification d'histogramme.
- La réduction du bruit ou lissage.
- Le rehaussement de contraste.

1.5.2. Modification d'histogramme

On cherche à améliorer l'image en lui appliquant une transformation ponctuelle d'intensité, de façon à conserver les contrastes relatifs entre régions (une région claire sur un fond sombre apparaîtra plus claire que le fond dans l'image transformée). L'image est considérée ici, soit comme un signal déterministe, soit comme un ensemble de réalisations d'une variable aléatoire.

1.5.3. Réduction du bruit

Le transfert de l'image d'un objet jusqu'à l'ordinateur lors de l'acquisition des images se produit avec un certain bruit.

La méthode la plus simple pour augmenter le rapport signal/bruit consiste à appliquer le principe des analyseurs multicanaux, c'est à dire effectuer plusieurs acquisitions de l'image (n par exemple): ce qui revient à effectuer plusieurs sommations du signal. Le bruit n'apparaissant statistiquement jamais au même endroit, sera uniformément réparti, alors que le signal, apparaissant toujours au même endroit, sera amplifié d'un facteur racine carrée de n.

1.5.4. Amélioration par filtrage

L'amélioration de l'image est essentiellement obtenue par ce que l'on appelle une opération de filtrage. Il existe un grand nombre de filtres possibles, et à quelques exceptions près, on peut les classer en 2 grandes catégories :

- Les filtres linéaires
- Les filtres morphologiques

1.5.4.1. Les filtres linéaires

L'utilisation des filtres linéaires provient de l'extension des méthodes mises au point pour le traitement du signal, au traitement des images numériques.

a). **Filtre Moyenneur :** Le niveau de gris du pixel central est remplacé par la moyenne des niveaux de gris des pixels environnants. Son noyau est :

$$\frac{1}{9} * \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

Nouvelle valeur

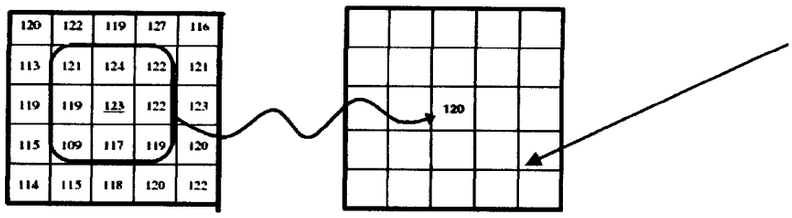


Figure 1.8 : principe de filtre moyenneur

b). Filtre de Smooth

Dans ce filtre, les coefficients sont choisis avec soins, permettant ainsi un traitement moins grossier de l'image. Les coefficients du noyau pour u
n filtre 3×3sont:

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

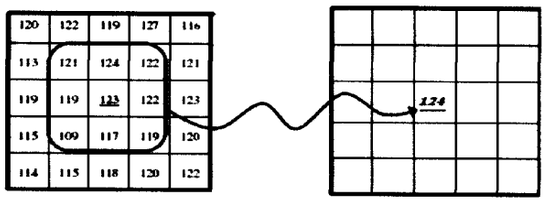


Figure 1.9 : principe de filtre de Smooth

Le gros avantage de ces filtres est leur facilité de conception et d'implémentation mais, ils ne peuvent être utilisés pour des travaux trop fins (la détérioration des contours qu'ils induisent par exemple, empêchera une segmentation fine des images).

Ces limitations ont donc conduit à la conception de filtres non linéaires présentés dans la partie suivante.

1.5.4.2. Filtres non linéaires (morphologies)

Les filtres morphologies s'intéressent à l'extraction d'ensembles de pixels constituant des formes dans une image par comparaison avec des éléments structurants de forme choisie jouant le rôle de filtres spatiaux.

a). Filtre médian : Le niveau de gris du pixel central est remplacé par la valeur médiane de tous les pixels de la fenêtre d'analyse.

La figure suivante illustre le fonctionnement d'un filtre médian de fenêtre d'analyse 3*3 sur un exemple:

$$\begin{pmatrix} 1 & 4 & 5 \\ 3 & 7 & 9 \\ 2 & 8 & 6 \end{pmatrix} \longrightarrow \begin{pmatrix} * & * & * \\ * & 5 & * \\ * & * & * \end{pmatrix}$$

Les niveaux de gris sont classés par ordre croissant : 1<2<3<4<5<6<8<9<10. La valeur médiane 5 est choisie.

L'opération de classement est coûteuse en tant de calcul.

b). Filtre Mini-Médian : le principe c'est d'appliquer le principe de filtre médian seulement sur les pixels infectés par un bruit blanc ; dans ce cas, on estime que le niveau de gris du pixel bruité est dans l'intervalle [min, 255].

La valeur min ∈ [0, 255], si min=0 alors on est dans le cas du filtre médian (c'est-a dire applique le filtre médian sur tout les pixels de l'image).

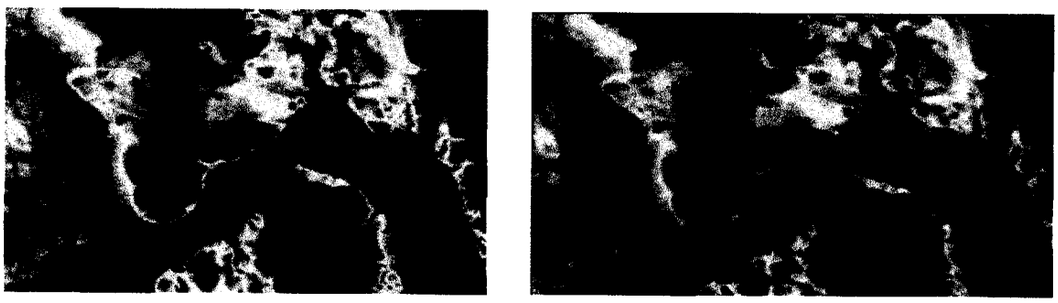


Figure 1.10 : Exemple de filtre mini médian

c). Le filtre maximum : on applique le même traitement de filtre médian mais la valeur de pixel de centre va être remplacée par le maximum.

Exemple: $\begin{pmatrix} 1 & 4 & 5 \\ 3 & 7 & 9 \\ 2 & 8 & 6 \end{pmatrix} \longrightarrow \begin{pmatrix} * & * & * \\ * & 9 & * \\ * & * & * \end{pmatrix}$

1.5.5. Rehaussement de contraste

L'opérateur de rehaussement de contraste devra, si possible, réduire le bruit dans les zones stationnaires éviter les phénomènes de dépassement. Ce type d'opérateur est réalisable au moyen de méthodes linéaires ou non linéaires.

1.5.6. Segmentation [7]

On est conduit pour analyser une image à la partager en un certain nombre de domaines connexes (unis): ce processus joue un rôle de plus en plus important en traitement numérique des images.

La segmentation consiste a partitionner l'image étudiée en régions disjointes avec

Des couleurs homogènes. La segmentation peut être basée sur différents critères Comme la couleur, la texture ou les niveaux de gris d'une image.

1.5.7. Visualisation

Tout système de traitement d'image est doté d'un dispositif de visualisation qui permet l'affichage des images.

L'utilisation de différents types de restituteurs permet de transformer le signal numérique qui est la matrice image en un signal analogique visible par l'œil de l'observateur. Pour cela, différents types de supports peuvent être employés : moniteur vidéo, clichés photographiques, impression sur papier. Dans tous les cas et pour chaque échantillon de l'image numérique, on recrée un nouvel élément d'image ou un nouveau pixel dont on choisit la forme de façon à reconstituer une image analogique qui soit la plus proche possible de l'image avant numérisation compte tenu des erreurs introduites lors de l'acquisition, de la numérisation et de la transmission.

1.6. Conclusion

L'analyse d'image touche à l'heure actuelle de nombreux domaine, avec des objectifs aussi variés. Le but du traitement d'images est à la fois simple dans son concept et difficile dans sa réalisation. Simple en effet, puisqu'il s'agit de reconnaître des objets que notre système visuel. Difficile cependant pour l'application visée et ceci indépendamment de la qualité de l'image. L'analyse d'image s'est donc dotée d'outils et de méthodes puissantes issues de domaines aussi variés que les mathématiques, le traitement du signal, ou l'informatique.

Le prochain chapitre sera consacré sur les techniques de segmentation des images.

Chapitre 2

Segmentation des images

2.1. Introduction

L'analyse d'une image cherche à passer d'une grande quantité d'informations de bas niveau (i.e. les pixels), à une description de haut niveau pouvant conduire à une interprétation de cette image. La segmentation représente souvent la première étape à cette analyse.

Segmenter une image, c'est extraire les objets qui la constituent en vue de décrire leurs relations et mesurer les paramètres qui les caractérisent.

Dans une image, un objet est constitué de l'ensemble de pixels connexes possédant une propriété commune et que ne possèdent pas les objets voisins. Cet ensemble est par définition entouré d'un contour fermé.

Pour éviter toute ambiguïté avec les objets physiques pouvant apparaître dans une image, nous préférons utiliser le terme de ***région*** au lieu d'objet. Un objet physique pourra alors être décrit par un ensemble de régions et ne pourra être reconnu comme tel que si l'on possède des connaissances a priori sur cet objet.

A l'opposé, une région sera telle à partir uniquement de propriétés d'images : niveaux de gris, couleur, texture.

De manière générale, la segmentation s'applique à des types très différents d'images et les techniques varient en fonction du domaine étudié. Il est en effet du probable qu'une technique générale de segmentation puisse fonctionner parfaitement sur toutes les images.

La littérature fait état à deux classes d'algorithme de segmentation :

Approche ***contour*** (frontière) et approche ***région***. Par la suite est née une autre approche appelée ***mixte*** basée sur la coopération des deux approches.

2.2. La segmentation [7]

Lorsqu'on dispose d'une image, on souhaite généralement distinguer des objets d'intérêt, c'est-à-dire trouver des caractéristiques locales adéquates permettant de les distinguer d'autres objets ou du fond. Cette opération est appelée segmentation.

La segmentation est l'opération entre le traitement d'image de bas niveau et l'analyse d'image. Après la segmentation, on sait quel pixel appartient à quel objet et on peut aussi analyser la forme des objets par divers opérateurs morphologiques.

On peut définir la segmentation comme étant une partition de l'image I en
Sous-ensembles R_i appelés régions, tels que :

$\forall$ i $R_i \neq \Phi$

$\forall$ i,j ; i $\neq$ j $R_i \cap R_j = \Phi$

$I = \cup R_i$

Fondamentalement, la segmentation est un processus qui consiste à découper une image en régions connexes présentant une homogénéité selon un certain critère, comme par exemple la couleur. L'union de ces régions doit redonner l'image initiale.
On regroupe généralement les algorithmes de segmentation en trois grandes classes:

- Segmentation basée sur les contours
- Segmentation basée sur les régions
- Segmentation mixte

2.2.1. Objectif de la segmentation

- Découper l'image en domaines correspondant aux objets (composantes) de l'image.
- Etiqueter chaque pixel de l'image (une étiquette par composante).

Exemple d'application de la segmentation

1. détection d'une tumeur dans une image médicale.
2. vidéosurveillance : objets, personnes mobiles/immobiles
3. détection des bords dans une image : bords/intérieurs
4. images satellites : {zones urbaines, champs, forêts, routes, mer}

2.2.2. Homogénéité

L'homogénéité est une information locale et correspond au caractère uniforme d'une région. Elle est composée de deux informations sur l'intensité : d'une part son écart-type, d'autre part ses discontinuités (norme du gradient). En chaque point l'écart type décrit le contraste de l'intensité dans le voisinage du point considéré, et la mesure de discontinuité indique le changement brusque de niveau de gris toujours dans un voisinage de ce point. Pour une représentation RGB, l'intensité ***I*** vérifie : ***I=(R+G+B)/3.***

2.2.3. Région

C'est une zone homogène dans l'image (niveau de gris, couleur, texture, gradient, etc.).

Soit I une image, dans une segmentation en régions → partition de I en K régions Rk, on a :

- Tout pixel appartient à une région.
- Aucun pixel n'appartient à plus d'une région.
- Cohérence des attributs de région.

2.2.4. Critère de similarité

La mesure de la similarité de deux pixels est le point clé des techniques de segmentation. Il convient donc de choisir avec soin la méthode de calcul de la similarité. L'approche la plus simple consiste à définir comme similaires des pixels qui sont visuellement proches. Pour une image en niveau de gris, il faut calculer la différence entre les valeurs. Si cette différence est inférieure à un certain seuil, les pixels seront jugés similaires. Cette méthode est équivalente à calculer la norme du gradient de l'image en chaque point et à seuiller le résultat obtenu.

2.2.5. La squelettisation

Appelée aussi amincissement, est une méthode s'effectuant sur une image binaire (2 niveaux; noir ou blanc), elle doit conserver trois propriétés sur l'image :

- Doit être une région amincie, d'un seul pixel de large.
- Les pixels du squelette doivent être proches du centre de la région originale.
- Les pixels du squelette doivent être connectés de façon à préserver les formes originales et leur direction.

Image à squelettiser. Image squelettisée.

Figure 2.1 : squelettisation d'une image [S5]

2.3. Les différentes approches de segmentation [3]

À ce jour, il existe de nombreuses méthodes de segmentation, que l'on peut regrouper en trois principales classes :

2.3.1. Approche contour

Cette approche est une technique non contextuelle qui ignore les relations pouvant exister entre les régions de l'image. Les pixels sont regroupés en fonction d'un attribut global
La recherche des contours dans une image numérique est un des problèmes les plus étudiés depuis l'origine des travaux sur l'imagerie numérique. Ceci est en grande partie dû à la nature très intuitive du contour qui apparaît très naturellement comme l'indice visuel idéal dans la plus grande partie des situations.

Le but des détections de contours est de détecter dans l'image les points de contour, les frontières entre les objets. La détection de contour est une étape préliminaire à de nombreuses applications de l'analyse d'images. Les contours constituent en effet des indices riches, au même titre que les points d'intérêts, pour toute interprétation ultérieure de l'image. Les contours dans une image proviennent des:

- Discontinuités de la fonction de réflectance (texture, ombre)
- Discontinuités de profondeur (bords de l'objet)

Un détecteur de contour passe par trois étapes figure (2.2) :

- **Lissage de l'image:** L'opération de lissage a pour objectif la réduction du bruit contenu dans l'image cette opération s'effectue par une convolution de l'image avec des filtres passe-bas linéaires.

- **La différentiation de l'image:** Un point de contour ou de transition peut être défini comme un point au voisinage duquel la fonction *f* de niveau de gris présent une variation de forte amplitude, si on considère la fonction *f* comme un signal monodimensionnel, la variation peut être détectée par le maximum d'une dérivée première, ou bien par le passage au zéro de la dérivée seconde. Alors l'intérêt de la différentiation qui calcule les dérivées nécessaires pour mettre en évidence les contours.

- **La correction:** Augmenter le rapport signal/bruit du détecteur par une suppression des faux points de contours et par la fermeture des trous dans les contours résultants.

Cette opération est secondaire car sa fonction peut être réalisée dans les étapes précédentes, le lissage élimine les bruits en préservant les contours, et la différentiation fournit ces contours, donc le problème de correction et résolut.

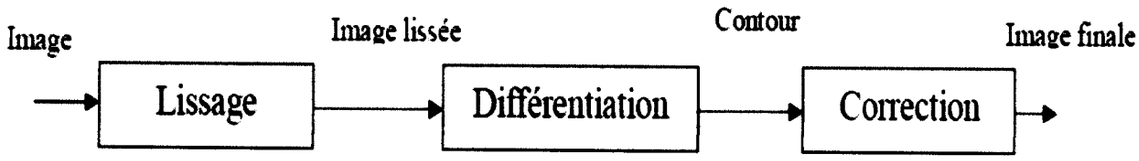


Figure 2. 2 : Schéma de fonctionnement d'un détecteur de contours.[S6]

Image originale Image traitée

Figure 2.3 : Exemple de détection de conteur[S4]

2.3.2. Approche Région [5]

La segmentation des images consiste à créer une partition de l'image en sous-ensembles homogènes appelés régions, selon un certain critère.

La segmentation d'une image I en régions $\mathbf{R_i}$ a pour but de regrouper des pixels connexes ayant des couleurs similaires, afin de constituer des régions de couleurs homogènes, ces régions étant dans la mesure du possible liées à un objet ou à une partie d'un objet représenté dans l'image.

Les pixels de chaque région doivent respecter des critères d'homogénéité et de connexité. L'homogénéité d'une région $\mathbf{R_i}$ est définie par un prédicat d'uniformité, noté $\mathbf{Pred(R_i)}$. Ce prédicat est vrai si R_i est homogène, faux dans le cas contraire.

Les régions doivent respecter les quatre conditions suivantes :

- $\mathbf{I} = U_i R_i$
- R_i est constituée de pixels connexes pour tout i
- $\mathbf{Pred}(R_i)$ = vrai pour tout i
- $\mathbf{Pred(R_i \cup R_j)}$ = faux pour tout $i \neq j$, Ri et Rj étant adjacents dans **I.**

La première condition implique que chaque pixel de l'image doit appartenir à une région R_i et que l'union de toutes les régions correspond à l'image entière. La deuxième condition est relative à la structure des régions. Elle définit une région comme un sous-ensemble de pixels connexes. La troisième condition exprime que chaque région doit respecter un prédicat d'uniformité. La dernière condition implique la non-réalisation de ce même prédicat pour toute réunion de deux régions adjacentes.

Le résultat de la segmentation est une image dans laquelle est attribuée à chaque pixel une étiquette correspondant à la région à laquelle il appartient.

Image originale Image segmentée

Figure 2.4 : Segmentation par approche région.[S9]

2.3.3. Quelques méthodes de segmentation par région [10]

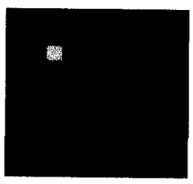
2.3.3.1. Les méthodes de type croissance de régions

La croissance de régions est une technique contextuelle ; elle prend en considération les attributs locaux des pixels. C'est une technique essentiellement ascendante. La croissance peut être guidée par différents critères. Lorsque l'homogénéité guide la croissance, un pixel dont le niveau de gris est proche de celui de la région courante, est sélectionné. L'évaluation porte généralement sur une mesure de dispersion locale des niveaux de gris.

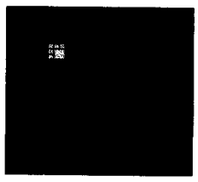
2.3.3.2. Segmentation par division de régions

L'approche segmentation par division de régions consiste à diviser l'image originale en régions homogènes au sens d'un critère donné. Ce processus est récursif et considère que la région initiale correspond à l'image à analyser. Si une région ne respecte pas un prédicat d'homogénéité, elle est divisée en quatre sous régions de taille égale. Chaque sous région est ensuite analysée. L'algorithme récursif s'arrête lorsque toutes les régions respectent le prédicat d'homogénéité.

A cause des divisions en quatre des régions, cette méthode est plutôt adaptée à des images carrées ayant un nombre de lignes et de colonnes égal à une puissance de deux, et dans lesquelles les régions sont de forme rectangulaire. D'autre part, cette méthode a tendance à faire apparaître des effets de blocs.



a. Image Originale



b. Image segmentée

Figure 2.5 : Exemple de segmentation.

2.3.3.3. Les méthodes de type division-fusion "split-merge" [5]

Après une étape d'initialisation, le processus de segmentation est itératif et alterne deux phases : une phase de division de toutes les régions non homogènes et une phase de fusion de toutes les régions adjacentes de sorte que la région résultante respecte toujours le critère

d'homogénéité. Ces méthodes font appel à la théorie des graphes, ainsi elles peuvent être classées selon la structure du graphe.

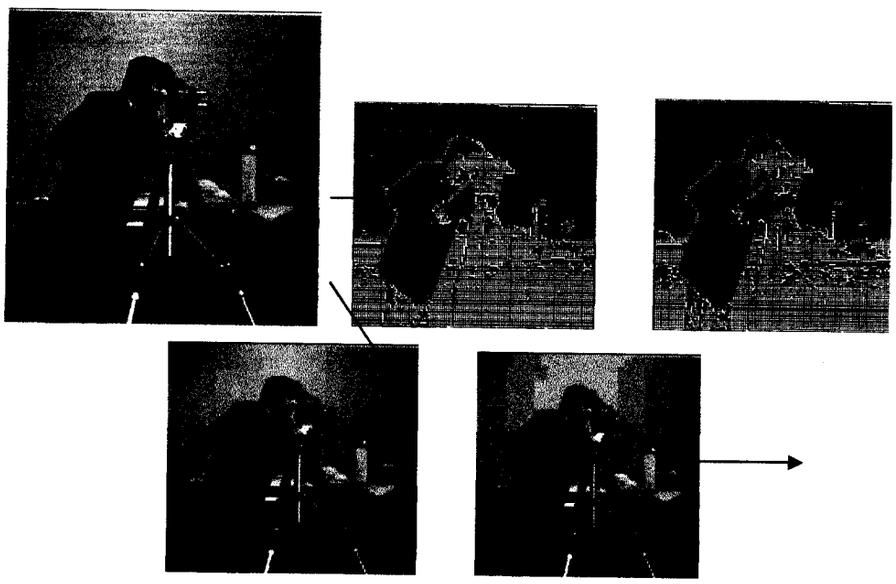


Figure 2.6 : Division et Fusion

Segmentation par les champs de Markov

Une image est considérée par le formalisme des champs de Markov comme une réalisation particulière d'un champ de variables aléatoires. Les champs de Markov sont très intéressants dans le sens où ils ne se limitent pas à la prise en compte individuelle des pixels car ils permettent de prendre en compte les relations de voisinage entre les pixels. Ainsi, la probabilité qu'un pixel appartienne à une classe dépend non seulement de sa couleur, mais aussi de celles de ses voisins.

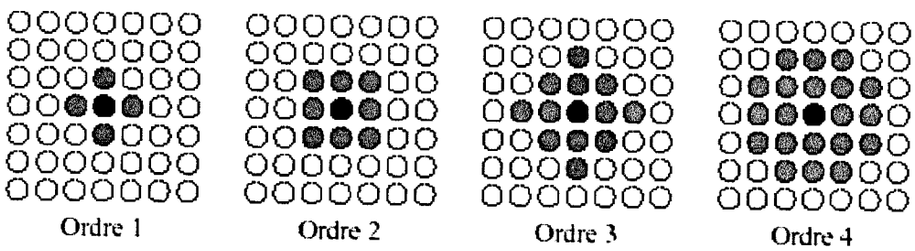


Figure 2.7 : voisinage d'un champ aléatoire

2.4. Les critères de choix de la technique de segmentation [8]

Il n'y a pas de méthode unique de segmentation d'une image. Le choix d'une technique est lié :

➢ ***A la nature de l'image*** :

Eclairage non homogène, reflets.

Présence de bruit, de zones texturées.

Contours flous, en partie occultée.

➢ ***Aux opérations situées en aval de la segmentation :***

Localisation, mesure, calcul 3D.

Reconnaissance des formes, interprétation.

Diagnostic, contrôle de qualité.

➢ ***Aux primitives à extraire :***

Contours, segmentation de droite, angles.

Région, formes.

Textures.

2.5. Conclusion

Dans ce chapitre nous avons abordé le problème de segmentation d'image, et les différentes techniques utilisées pour résoudre ce problème de segmentation.

La première famille de méthodes de segmentation utilise une approche de contour (ou frontières). Dans cette approche, on s'intéresse aux variations des niveaux de gris des pixels. Ainsi plusieurs sous-sections sont exposées.

D'autre part une segmentation basée sur une approche région était le principe de la deuxième famille des méthodes qui aboutissent directement à une partition de l'image, chaque point étant affecté à une région unique, car cette approche est basée sur les caractéristiques propres des constituants. Ce modèle nécessite des informations a priori sur les constituants pour pouvoir les différencier.

Alors, après cette vision disant au moins générale sur les méthodes de segmentation, on peut en déduire ou dire qu'il n'existe pas de méthodes idéales à la segmentation; mais l'objectif est d'arriver à une approximation meilleure des résultats quelque soit de la nature de l'image étudiée.

Le prochain chapitre sera consacré sur l'analyse des textures.

Chapitre 3

Analyse des textures

3.1. Introduction

La texture, qui est presque omniprésente dans les images, joue donc un rôle important en analyse d'images non seulement dans les applications de segmentation mais aussi de classification et de caractérisation. Bien qu'elle ait intéressé de nombreux chercheurs et que de nombreux travaux aient été publiés ces dernières années, la texture continue à susciter l'intérêt des chercheurs.

L'analyse de textures tend à occuper à l'heure actuelle une place de plus en plus grande dans les activités de recherches en traitement d'image. Ceci s'explique par le fait que la texture renferme une bonne part de l'information utile au système visuel humain pour analyser, segmenter et donc interpréter une image.

La plupart des études entreprises conduisent à des applications très diverses comme la classification de terrains dans les images aériennes, l'aide à l'interprétation d'images radiographiques et échographiques ou bien encore l'analyse d'images microscopiques.

3.2. Définition [7]

Le terme « texture » désigne l'information de type spatial, relative à l'état de surface de chacune des zones homogènes d'une image, chaque zone étant délimitée par une structure de signal de type monodimensionnel appelé contour.
On distingue dans la littérature, deux approches très différentes, voire opposées, visant à définir la notion de texture

- **Une approche dite syntaxique ou structurelle**, dans laquelle la texture est vue comme la répétition spatiale d'un motif de base {ou primitive} dans différentes directions de l'espace.

- **Une approche probabiliste**. dans ce cas, l'accent est mis sur l'aspect anarchique mais homogène de la texture.

Une texture est une région d'une image pour laquelle il est possible de définir une fenêtre de dimensions minimales, telle qu'une observation a travers de celle-ci se traduit par une perception (impression) visuelle identique pour toutes les translations possibles de cette fenêtre à l'intérieur de la région considérée.

3.3. Types de textures [8]

En pratique, on distingue deux grandes classes de textures, qui correspondent à deux niveaux de perception:

3.3.1. Les macro textures

Les macro textures présentent un aspect régulier, sous formes de motifs répétitifs spatialement placés selon une règle précise (exemple: peau de lézard, mur de brique) donc une approche structurelle déterministe.

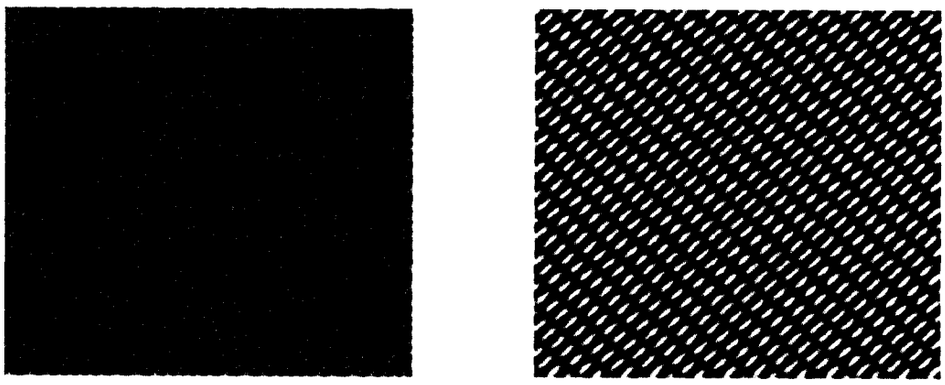


Figure 3.1 : Exemples de textures déterministes[S10]

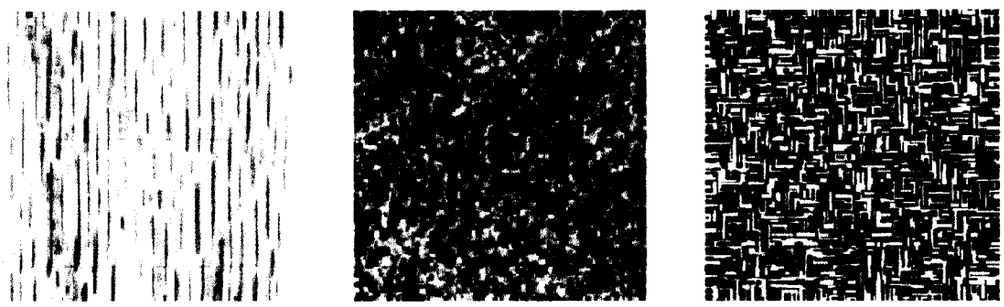
Une région texturée est constituée par un réseau bidimensionnel répétant le motif originel selon une direction et une période particulière.

La description du motif élémentaire, les dimensions du réseau et son orientation suffisent alors à décrire complètement la texture. Cette définition ne convient qu'à des textures parfaitement régulières que l'on rencontre rarement dans la réalité. Il est évident que l'image de la trame d'un textile et celle d'un champ de blé ne revêt pas le même caractère de régularité géométrique dans la disposition des motifs texturaux.

3.3.2. Les micro textures

Les micros textures présentant des primitives "microscopiques" distribuées de manière aléatoire (exemple: Sable, laine tissée, herbe) d'ou une approche probabiliste cherchant à caractériser l'aspect anarchique et homogène.

Pour des motifs et des arrangements irréguliers, la texture est dite stochastique. Une texture de ce type peut être considérée comme une réalisation d'un champ aléatoire bidimensionnel (Figure 3.2).



a. Texture " Bois " b. Texture " Herbe " c. Texture " markovienne "

Figure 3.2 : Exemples de textures stochastiques naturelles et de synthèse.

Les statistiques des niveaux de gris des pixels de l'image ou les paramètres du modèle stochastique obtenus à partir d'une réalisation fourniront les caractéristiques de la texture.

Brodatz a établi un catalogue de texture à des fins de synthèse d'images texturées (Brodatz, 1966). Ce catalogue constitue aujourd'hui une référence en matière de base de test pour les chercheurs travaillant sur la texture. Il s'agit de 112 textures (sable, herbe, bulles, eau, bois, etc.) ayant chacune seize variantes différentes. Ces images sont classiquement utilisées pour valider les différentes méthodes d'analyse de texture. (Figure-3.3).

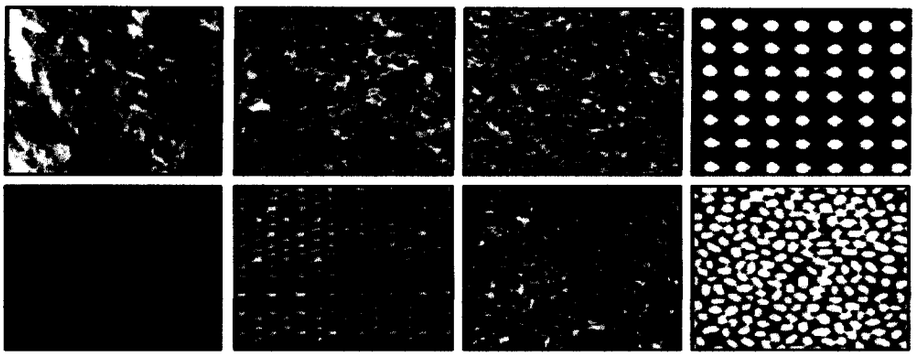


Figure 3.3 : Exemples de textures issues de l'album de Brodatz

3.4. Les méthodes utilisées pour caractériser les textures [1]

Dans la littérature on trouve toute une panoplie de méthodes d'analyse des textures répondant a des besoins différents mais surtout soumises a des contraintes plus au moins sévères. Il nous a donc fallu faire un choix parmi cet ensemble selon certains critères liés à l'application.

3.4.1. La méthode des histogrammes

Cette méthode consiste à extraire l'information importante de l'image a partir des histogrammes de luminance. Chaque histogramme obtenu sur une fenêtre de taille n*n (=n^2 pixels) est défini comme un vecteur caractérisé par sa norme (ou module) et sa phase. Les luminances étant quantifiées sur r niveaux de gris.

3.4.2. La méthode des extréma

Dans sa méthode, Michel utilise la fréquence relative des extréma dans les niveaux de gris c'est-à-dire la fréquence relative des pics de luminance comme mesure principale pour caractériser la texture

L'algorithme développé comporte deux étapes ; la première étape correspond à un processus de lissage destiné à éliminer les pics de luminance de faible amplitude afin de ne garder que les extréma principaux. La deuxième étape correspond à la détection et au comptage des extréma principaux.

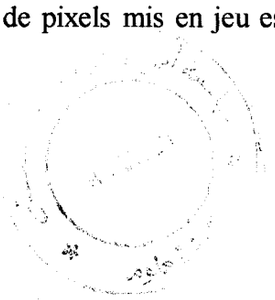
3.5. Analyse de la texture [2]

L'analyse de texture regroupe un ensemble de techniques permettant de quantifier les différents niveaux de gris présents dans une image en termes d'intensité et de distribution dans le but de calculer un certain nombre de paramètres caractérisant la texture à étudier.

Comme il existe deux grandes classes de textures, l'analyse peut se faire selon deux classes de méthodes: les méthodes structurelles et les méthodes statistiques.

Les méthodes structurelles permettent de décrire la texture en définissant les primitives et les "règles" d'arrangement qui les relient. Elles sont donc applicables sur les textures aléatoires. Alors que les méthodes statistiques étudient les relations entre un pixel et ses voisins et définissent des paramètres discriminants de la texture en se basant sur des outils statistiques.

Généralement, elles sont utilisées pour caractériser des structures fines, sans régularité apparente. Plus l'ordre de la statistique est élevé et plus le nombre de pixels mis en jeu est important.



3.6. Perception et analyse visuelle d'une texture [S11]

3.6.1. Vision et perception par l'œil

L'œil humain distingue en moyenne 16 niveaux de gris différents du noir au blanc. Un œil peu exercé en voit moins .alors qu'un professionnel peut en distinguer jusqu'à 20. En revanche, il est capable d'apprécier des différences de niveaux voisins de 2% (contraste).
Une texture pourra être perçue de différentes façons suivant la manière dont le cerveau traitera les informations visuelles.

Il existe deux types de perception : celle qui ne demande pas d'effort et celle qui demande de scruter l'image plus longtemps. Sur le schéma ci-dessous, il est impossible de voir spontanément que l'image de gauche est formée d'une seule spirale alors que celle de droite en contient deux.

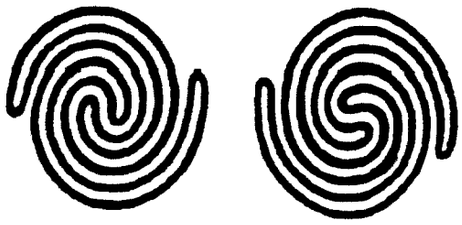


Figure 3.4 : Exemple de la limite de la perception " spontanée ".

3.7. Méthodes d'analyse de texture [7]

Le but de l'analyse de texture est de formaliser les descriptifs de la texture par des paramètres mathématiques qui serviraient à l'identifier. Dans ce sens, les critères visuels qui ont été retenues pour la texture sont: le contraste, l'orientation, la forme, la finesse et la régularité.

3.7.1. Les méthodes structurelles

Ces méthodes tiennent compte de l'information structurelle et contextuelle d'une forme et sont particulièrement bien adaptées aux textures macroscopiques. Les étapes d'analyse sont d'abord l'identification des éléments constitutifs, puis la définition des règles de placement. Les deux structures les plus importantes sont les structures de graphe et les structures syntaxiques.

3.7.2. Les méthodes statistiques

Dans ces méthodes la texture est considérée comme la réalisation d'un processus stochastique stationnaire. Des parameters statistiques sont estimés pour chaque pixel de l'image. Suivant la modalité des images à étudier, la signature la plus discriminante de la texture est à rechercher soit dans des méthodes qui exploitent directement les propriétés statistiques de la texture (modèle de Markov), soit dans des méthodes qui exploitent les propriétés statistiques a partir d'un plan transformé dans lequel on réécrit l'image de texture (des filtres numériques).

3.7.3. Les méthodes basées sur l'étude des formes

Ces méthodes se trouvent au croisement de la reconnaissance des formes, de la caractérisation de défauts et de l'analyse macro textural. Les régions textuelles de l'image épousent des formes particulières et peuvent être caractérisées par des paramètres dits de formes.

3.7.4. Les méthodes spatio-fréquentielles

Dans ces méthodes les représentations spatio-fréquentielles préservent à la fois les informations globales et locales donc elles sont bien adaptées aux signaux quasi périodiques. En effet, les textures sont des signaux quasi périodiques qui ont une énergie fréquentielle localisée. Ces méthodes permettent de caractériser la texture à différentes échelles.

3.8. Domaines d'application de l'analyse de texture [S13]

L'analyse de texture est utilisée dans des domaines de plus en plus variés. La caractérisation, la segmentation des images ou la reconnaissance des formes représentent ses plus importantes applications.

☑ **En segmentation :** L'analyse de texture est souvent employée comme un moyen pour finaliser la segmentation d'une image. En effet, dans le cas des images naturelles, la seule étude de la distribution des niveaux de gris est insuffisante pour caractériser les zones homogènes. C'est le cas pour les images multi spectrales réalisées en télédétection ou les estimateurs de textures permettent de différencier un champ, d'une forêt, d'une ville..... C'est le cas aussi en imagerie médicale ou ces mêmes estimateurs permettent de faire la distinction entre tissus sains et tissus pathologiques (détection de lésions, dépistage de pathologies).

☑ **En compression d'images** : les attributs texturaux permettent de représenter l'image par un nombre minimal de paramètres et de ce fait permettent une reconstitution de l'information avec un minimum d'erreur.

☑ **En restauration :** La caractérisation de texture à l'aide de paramètres pertinents permet également la restauration d'une partie dégradée ou manquante dans une image en la remplaçant par une version synthétique générée a partir du modèle textural élaboré.

☑ **En infographie :** Dans le domaine de l'infographie ou de l'audiovisuel, la synthèse de texture découle naturellement de l'analyse et conduit à son utilisation pour le réalisme, l'art, le design.

☑ **En contrôle non destructif :** l'analyse de texture est largement utilisée pour l'inspection des surfaces en contrôle de qualité (produits industriels, matériaux, produits alimentaires, détection de défauts.).

3.9. Segmentation et détection de contours sur des images texturées [5]

En général les statistiques d'ordre un et deux sont suffisantes pour classifier les micro-textures, mais le problème est plus ardu dans le cas de macro-textures. Pour caractériser les micro-textures, un ensemble de caractéristiques et de textures classiques et des statistiques d'ordre un et deux : la moyenne, la variance et des covariances sont utilisées. Ensuite de nouvelles caractéristiques basées sur la détection de motifs pour les macro-textures ont été construites.

Chacun des attributs classiques est supposé gaussien, et les paramètres du modèle sont estimés sur des échantillons.

3.9.1. Modèle de contours

Nous nous intéressons ici à un problème plus simple que celui d'une détection complète de frontières de zones texturées, avec fermeture des contours. En effet, étant donné que nous obtiendrons finalement une segmentation de l'image, nous en déduirons immédiatement les frontières finales. Cette première étape de détection de contours est destinée à extraire des informations précises et relativement sures sur la position des éléments de contours. Cependant, puisque nous avons à faire *a priori* à des images fortement texturées, le problème n'est pas classique.

Deux fenêtres sont associées à chaque site contour.

La taille de ces fenêtres peut dépendre de la présence de macro-textures. L'algorithme permet d'utiliser deux tailles de fenêtres, l'une pour les micro-textures, et l'autre pour les macro-textures. Le choix de la taille de cette fenêtre se fait de la manière suivante :

- Tout d'abord, déterminer si un motif de macro-texture est présent dans la fenêtre de grande taille.

- Si le cas, il faut alors utiliser cette résolution, sinon, on utilise des fenêtres de petite taille correspondant à des micro-textures.

3.9.2. Modèle de segmentation [9]

Pour l'algorithme de segmentation, l'idée de base est de choisir en chaque site l'étiquette la mieux adaptée au vecteur d'attributs de texture observée. Il nous faut donc définir autour de chaque site une fenêtre dans laquelle seront calculés les attributs. En général, ce type de méthode induit une difficulté importante : à la frontière entre deux textures, une fenêtre contient un mélange des deux textures, si bien que les attributs de texture deviennent non significatifs.

Les éléments de contours peuvent être utilisés afin de pallier cet inconvénient majeur. En effet, ils permettent de déformer la fenêtre glissante à l'approche d'une frontière. D'autre part, il est aussi possible d'utiliser l'information de contours afin de stopper l'effet régularisation de l'énergie au travers d'une frontière.

- *Notations*

• $a = (a_s)$: niveaux de gris observés, s est le site pixel.

• M est le nombre total d'échantillons de textures (ou d'étiquettes), et l'indice k représente le $k^{éme}$ échantillon (ou étiquette).

• F_s : fenêtre d'observation centrée en s. La taille de F_s dépend de la présence de macro-textures.

• $A = (\lambda_s)$: configuration d'étiquettes qui définit la segmentation.

• V_s : voisinage de s (les huit plus proches voisins en général).

• l est la fonction indicatrice : $l_{\lambda s} = \lambda_r = 1$ si $\lambda_r = \lambda_s$ et 0 sinon.

• α_s = vecteur attribut calculé sur F_s.

• $\alpha_s^{(m)}$ est le $m^{éme}$ attribut. Cet attribut est calculé dans la fenêtre F_s qui sera tronquée suivant le nombre d'éléments de contours. Le calcule de $\alpha_s^{(m)}$ varie suivant deux critères :

Le premier est la taille de F_s :

S'il n'y a pas de motif de macro-texture autour de s, alors F_s est de taille correspondant à la résolution des micro-textures (la plus petite taille est de l'ordre d'un carré de 5 à 7 pixels de côté). Sinon F_s est de la taille correspondant à la résolution des macro-textures (de l'ordre d'un carré de 15 à 25 pixels de côté).

Le second critère est la présence d'éléments de contours :

S'il y a suffisamment d'éléments de contours dans F_s, alors nous calculons la droite la mieux adaptée (au sens des moindres carrés) aux contours observés, et nous coupons la fenêtre par cette droite, sinon, les attributs sont calculés sur la fenêtre entière.

3.10. Un algorithme supervisé de segmentation d'images texturées [8]

Cette approche implique que l'on dispose d'échantillons de chacune des textures présentes.

3.10.1. Modèle de texture

La texture joue un rôle fondamental dans la description et la classification des divers motifs dans les images. Beaucoup des images naturelles peuvent être considérer comme un ensemble de textures homogènes, qui sont en ensemble avec leurs frontières, fournirant une description raisonnable de l'image analysée. Mais ce n'est pas facile de produire un modèle mathématique de texture parce qu'il existe ainsi plusieurs types de textures naturelles ou artificielles, et chaque texture a ses attributs.

Choisir un modèle de texture revient à définir la probabilité conditionnelle P(a/ λ) d'avoir la réalisation a sachant qu'on a la texture l en tout site (λ est une réalisation où $\lambda_s = l \ \forall \ s \in S$). Comme A est un champ de Markov sera vue dans le prochain chapitre, on peut écrire :

$$P(a/\lambda) = \frac{1}{Z^{(l)}} \exp\left(-U^{(l)}(a)\right)$$

$$\text{Où } Z^{(l)} = \sum_{a} \exp\left(-U^{(l)}(a)\right)$$

Nous utilisons, pour ce modèle, uniquement des cliques d'ordre deux appartenant au système de voisinage défini par les 48 plus proches voisins d'un site.

3.11. Principe de la Segmentation d'images texturées

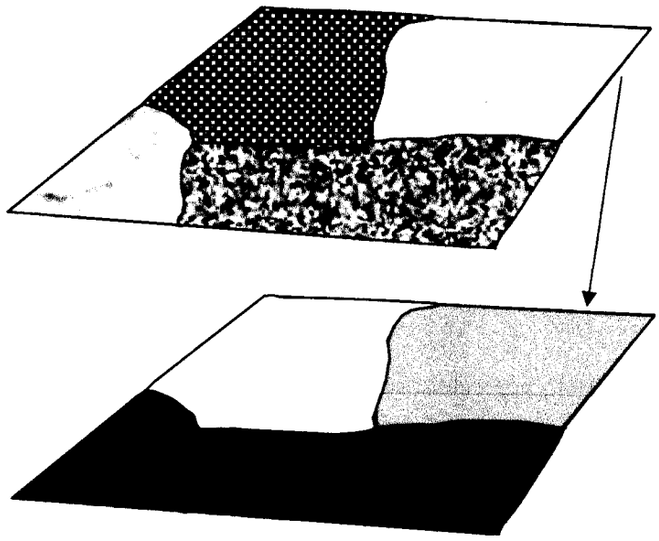


Figure 3.5 : illustration de segmentation d'une image texturée

1. Estimation du nombre de régions et de leurs paramètres (attributs).

2. Estimation des zones (régions et frontières).

Ces estimations peuvent être réalisées par l'optimisation de critères obtenus directement à partir des définitions de modèles stochastiques.

3.11.1. Modélisation stochastique

Image à segmenter = une réalisation d'un champ aléatoire non-stationnaire sur I (mélange de textures).

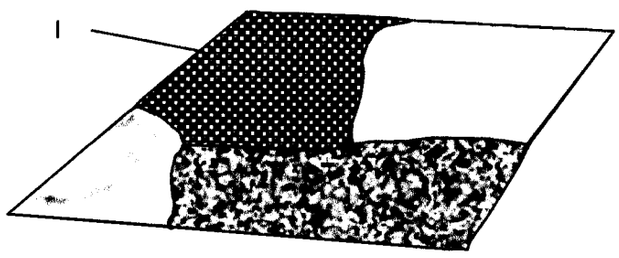


Figure 3.6 : Champ original texturé

$P(Y/\ y)$ $y= \{y_{i,j} = y_s\}_{s\ -\ (i,j) \in [0,\ T[\ \times\ [0,\ L[}$

$y_s \in R.$

Champ des régions = champ aléatoire de répartition de régions.

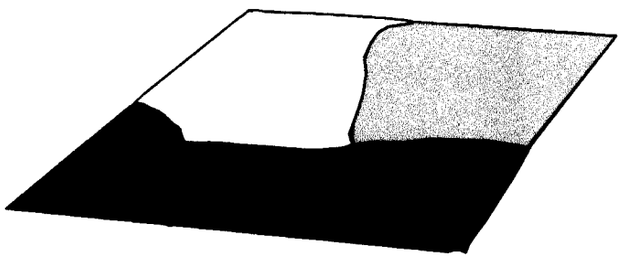


Figure 3.7 : Champ segmenté (à obtenir)

$P(X=x)$ $x= \{x_{i,j} = x_s\}_{s\ -\ (i,j) \in [0,\ T[\ \times\ [0,\ L[}$

$x_s \in [0,\ N_c]$ N_c : nombre de textures (classes).

3.11.2. Modélisation du champ de région

Hypothèse markovienne :

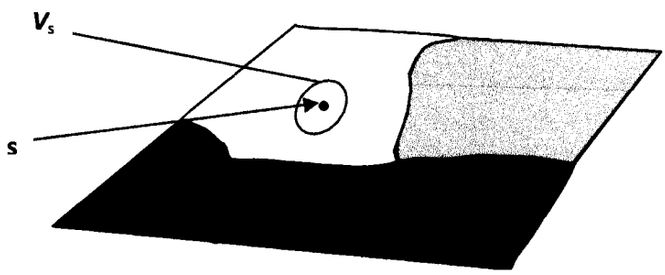


Figure 3.8 : Système de voisinage d'un site s

$V_s = \{s` \in I\ /\ s` \neq s \text{ et } s \in V_s\}$

$P(Xs= x_s/X- \{Xs\})= P(X_s= x_s/\ X_{Vs}= x_{Vs})$

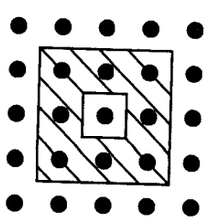


Figure 3.9 : exemple de voisinage

Loi de Gibbs $$p(X = x) = \frac{e^{-U(x)}}{Z}$$

Z : constante de normalisation sur l'ensemble des réalisations de X.

$$-\ln P(X = x) = U(x) + \ln(Z)$$

Le terme $U_2(x)$ s'exprime en fonction de potentiels associés à ces cliques :

$$U_2(x) = \sum_{c \in C} V^c(x)$$

3.12. Conclusion

Dans ce chapitre, nous avons parlés sur l'analyse des textures, nous avons présenté les concepts suivants : notions, types, modèles, méthodes d'analyse, attributs et les domaines d'applications de textures, et nous avons vu que la segmentation est l'uns des domaines d'application les plus importants de l'analyse de texture.
Le principe de la segmentation de texture consiste à choisir un modèle de texture.

Le prochain chapitre, sera consacré pour l'étude de la méthode de champ de markov appliquer sur la segmentation des images.

Chapitre 4

Segmentation d'image par champ Markov

4.1. Introduction

Les champs de Markov sont utilisés depuis maintenant une dizaine d'années en traitement d'images et font désormais partie des techniques de base de cette discipline. Nés à l'origine dans le cadre de la physique statistique pour étudier les phénomènes de transition de phase, ils sont rapidement appliqués aux réseaux bidimensionnels que constituent les images.
L'utilisation des champs de Markov est devenue populaire en traitement d'images depuis la publication de Geman et Geman en 1984.
Les techniques markoviennes sont utilisées pour améliorer la qualité de la segmentation, il existe plusieurs domaines abordés lors de la segmentation et utilisent des modélisations stochastiques plus précisément :

☑ La modélisation de textures.

☑ Les déformations observées entre une image idéale et une image bruitée.

☑ Les critères que l'on est amené à choisir, pour décrire la segmentation que l'on souhaite calculer est généralement complexes.

L'image numérique se présente et se manipule sous forme d'un tableau bidimensionnel (ou n dimensionnel) d'une variable entière quantifiée. L'information véhiculée par cette représentation va en réalité bien au delà de la seule donnée des niveaux de gris en chaque site, et la description d'une image se fait en termes de zones, contours, structures définis par les contrastes, textures, etc. qui peuvent être présents dans l'image. Le niveau de gris en un site n'est donc souvent pas significatif en lui-même, mais dans ses relations et interactions avec les pixels voisins.

Cette propriété des images, à savoir les interactions locales entre niveaux de gris voisins pour définir les différentes régions de l'image, va nous permettre d'utiliser un formalisme markovien dans de nombreux traitements, qu'il s'agisse de restauration, de segmentation et plus tard d'analyse complète des images. Le principe est de définir des énergies locales entre groupes de sites reflétant les interactions entre niveaux de gris. L'énergie globale est alors reliée à la probabilité d'apparition de l'image dans le cadre des champs de Gibbs.[3]

Dans ce chapitre, nous introduisons tout d'abord de façon intuitive la notion d'énergie locale avant de définir plus formellement un champ de Markov et d'énoncer le théorème d'équivalence entre champs de Markov et champs de Gibbs. Les algorithmes

d'échantillonnage d'un champ de Markov (échantillonneur de Gibbs) sont ensuite présentés, ainsi que les modèles markoviens les plus courants.

4.2. Description et modélisation probabiliste de l'image[10]

L'image est formée d'un ensemble fini S de sites si correspondant aux pixels.
S est donc essentiellement un réseau discret fini, partie de Z^d, si on note d la dimension de l'espace (2 le plus classiquement, 3 pour les volumes, etc.). A chaque site est associé un descripteur, représentant l'état du site et qui peut être son niveau de gris, une étiquette, ou une information plus complexe, et prenant ses valeurs dans E.

La notion d'interactions locales nécessite de structurer les relations spatiales entre les différents sites du réseau.

Pour ce faire, on munit S d'un système de voisinage V défini de la façon suivante:

$$\mathcal{V}_s = \{t\} \text{ tels que} \begin{cases} \cdot & s \notin \mathcal{V}_s \\ \cdot & t \in \mathcal{V}_s \Rightarrow s \in \mathcal{V}_t \end{cases}$$

A partir d'un système de voisinage, un système de cliques peut être déduit : une clique est soit un singleton de S, soit un ensemble de sites tous voisins les uns des autres. En fonction du système de voisinage utilisé, le système de cliques sera différent et fera intervenir plus ou moins de sites comme illustré sur la figure 4.1. On notera C l'ensemble des cliques relatif à V, et C_k l'ensemble des cliques de cardinal k.

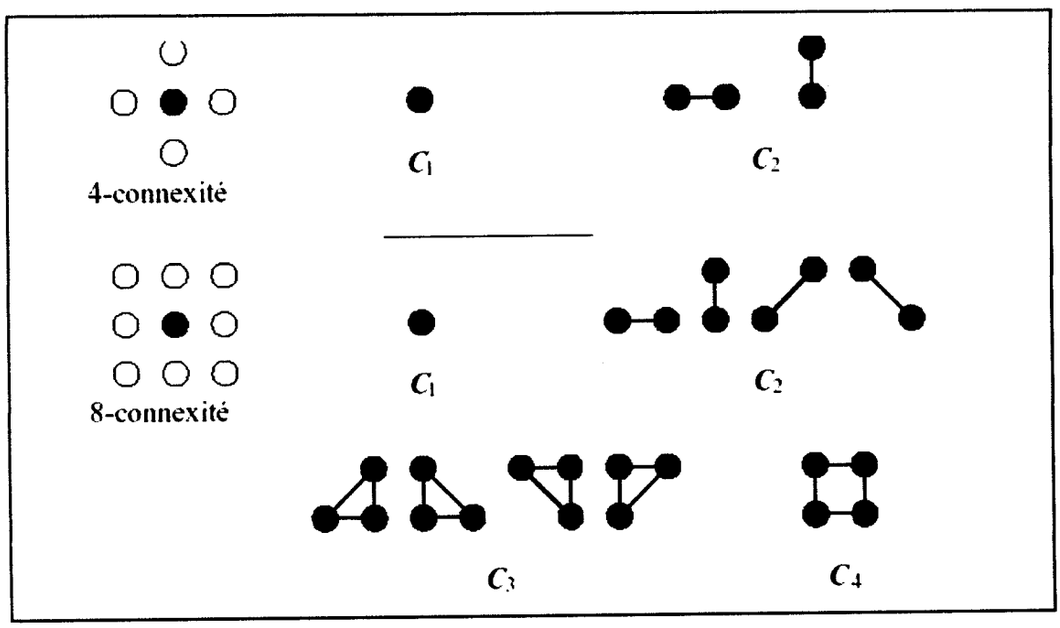


Figure 4.1 : Les cliques associées à divers systèmes de voisinage en dimension d = 2

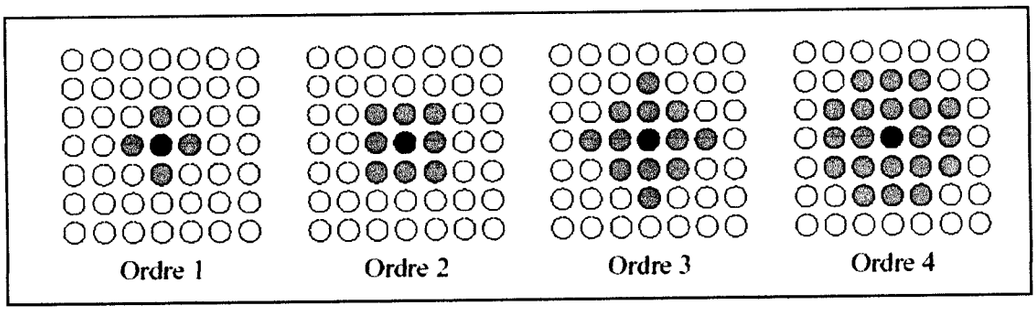


Figure 4.2 : Voisinage d'un champ aléatoire markovien

Les interactions locales entre niveaux de gris (ou descripteurs) de sites voisins peuvent alors s'exprimer comme un potentiel de clique. Soit c une clique, on lui associe le potentiel Uc dont la valeur dépend des niveaux de gris (ou descripteurs) des pixels constituant la clique. En poursuivant ce raisonnement, on peut définir l'énergie globale de l'image comme la somme des potentiels de toutes les cliques :

$$U = \sum_{c \in C} U_c$$

Et l'énergie locale en un site comme la somme des potentiels de toutes les cliques auxquelles il appartient :

$$U_s = \sum_{c \in C / s \in c} U_c$$

Nous avons jusqu'ici considéré le cas d'une image pour illustrer les notions de voisinage, de clique et de potentiel, mais le formalisme markovien se définit très généralement sur tout graphe. Soit un ensemble de sites S dénombrable (sommets du graphe), et une relation de voisinage, les cliques sont alors définies comme les sous-graphes complets du graphe. C'est l'utilisation de graphes plus généraux que ceux définis sur la grille de l'image qui permettent des traitements de plus haut niveau.[S15]

4.3. Modélisation probabiliste de l'image[11]

La définition des champs de Markov qui sera donnée dans la section suivante nécessite une modélisation probabiliste de l'image. Ainsi, l'image dont nous disposons va être considérée comme une réalisation d'un champ aléatoire. Soit s un site de l'image, on peut en effet lui associer une variable aléatoire X_s prenant ses valeurs dans E. Le niveau de gris x_s en s n'est ainsi qu'une réalisation de la v.a X_s. On définit alors le champ aléatoire $X = (X_s; X_t; ....)$ prenant ses valeurs dans $\Omega = E^s$. On trouvera aussi le terme de processus aléatoire pour X; en toute rigueur, "processus" devrait etre réservé au cas d'un ensemble d'indexation continu, et champ au cas discret.

Dans ce cadre probabiliste, l'image considérée est simplement une réalisation x du champ.
La probabilité globale de x, P(X = x), permet d'accéder en quelque sorte à la vraisemblance de l'image, et les probabilités conditionnelles locales d'une valeur en un site permettent de mesurer le lien statistique entre un niveau de gris et le reste de l'image. L'hypothèse markovienne permet d'évaluer ces quantités.

Notons que nous nous plaçons dans le cas ou E, l'espace de valeurs des descripteurs, est quantifié, ce qui nous permet de manipuler des probabilités.

4.4. Champs de Markov - Champs de Gibbs

4.4.1. Définition d'un champ de Markov[9]

Considérons x_s la valeur du descripteur prise au site s et $x^s = (x_t)_{t \neq s}$ la configuration de l'image excepté le site s. La définition d'un champ de Markov est alors la suivante :

> **X est un champ de Markov ssi la probabilité conditionnelle locale en un site n'est fonction que de la configuration du voisinage du site considéré**

Ce qui s'exprime de façon formelle par :

$$P(X_i = x_i \ / \ x^i) = P(X_i = x_i \ / \ x_j . j \in \mathcal{V}_i)$$

Ainsi, le niveau de gris en un site ne dépend que des niveaux de gris des pixels voisins de ce site. Cette hypothèse markovienne se justifié bien dans le cas de la plupart des images naturelles constituées de zones homogènes ou texturées ainsi que pour une large gamme d'images de synthèse. Plus généralement, une connaissance locale de l'image suffit souvent à réaliser son interprétation partielle et donc cette hypothèse markovienne sera souvent justifié sur des graphes plus globaux que le graphe des pixels.

Notons qu'en l'absence de contrainte sur le système de voisinage, tous les champs aléatoires peuvent être considérés comme markoviens à condition de prendre un voisinage suffisamment grand. L'intérêt de cette modélisation réside bien sûr dans le cas ou la propriété markovienne est vérifiée pour des voisinages restreints permettant des calculs rapides.

4.4.2. Equivalence champs de Markov - champs de Gibbs[7]

La modélisation markovienne prend toute sa puissance grâce au théorème que nous allons voir maintenant. En effet, celui-ci permettra d'accéder aux expressions des probabilités conditionnelles locales. Il nous faut au préalable définir un certain nombre de notions relatives aux mesures et champs de Gibbs.

4.4.3. Définition d'une mesure de Gibbs: La mesure de Gibbs de fonction d'énergie (ou d'Hamiltonien)[S17]

U : Ω→R est la probabilité P définie sur Ω par :

$$P(X = x) = \frac{1}{Z}\exp(-U(x))$$

$$U(x) = \sum_{c \in C} U_c(x)$$

Où C est le système de cliques associé au système de voisinage V de U[2].

$$Z = \sum_{x \in \Omega} \exp(-U(x)) \text{ est}$$

Est une constante de normalisation appelée fonction de partition de Gibbs.

La notation couramment utilisée pour U(x) est abusive car $U_c(x)$ ne dépend pas de l'ensemble de la configuration x mais seulement de x restreinte à la clique c ($U_c(x) = U_c(x_j,\ j \in c)$).

Nous pouvons maintenant définir le champ de Gibbs de potentiel associé au système de voisinage V:

C'est le champ aléatoire X dont la probabilité est une mesure de Gibbs associée au système de voisinage V, ce qui implique :

$$P(X = x) = \frac{1}{Z}\exp(-U(x)) = \frac{1}{Z}\exp\left(-\sum_{c \in C} U_c(x)\right)$$

L'énergie globale d'un champ de Gibbs possède donc la propriété de se décomposer sous forme d'une somme d'énergies locales, qui comme on le verra par la suite permettront d'accéder aux probabilités conditionnelles locales. Notons ici que plus une configuration d'un champ de Gibbs a une énergie faible, plus elle est probable.

Par exemple, si nous considérons un champ de Markov de voisinage 4-connexe, nous pouvons écrire l'énergie de la configuration x sous la forme :

$$U(x) = \sum_{c=(i)\in C_1} U_c(x_i) + \sum_{c=(i,j)\in C_2} U_c(x_i, x_j)$$

Si nous résumons les résultats précédents, la définition d'un champ de Markov passe par la définition de sa fonction d'énergie U. Celle-ci nécessite la définition d'un système de voisinage, qui définit alors le système de cliques, et de fonctions de potentiel associées aux cliques. [S19]

4.4.4. Modèle markovien gaussien [6]

Ce modèle est réservé aux images en niveaux de gris E = {0, ...255} et ne convient pas bien aux images d'étiquettes. Le voisinage est 4 ou 8-connexe et l'énergie est de la forme:

$$U(x) = \beta \sum_{c=(s,t)} (x_s - x_t)^2 + \alpha \sum_{s\in S} (x_s - \mu_s)^2$$

Le premier terme correspondant aux cliques d'ordre 2 est un terme de régularisation, qui favorise les faibles différences de niveaux de gris entre sites voisins pour $\beta > 0$. Le second terme peut correspondre à un terme d'attache aux données dans le cas où on possède une image de données extérieures. Des exemples de synthèse de modèles gaussiens sont montrés figure 4.3. Le rapport α/β pondère les influences respectives de l'attache aux données et de la régularisation, et les valeurs absolues des paramètres caractérisent le caractère plus ou moins piqué ou équi-réparti au contraire de la distribution.

Le modèle gaussien favorise des niveaux de gris proches pour des pixels voisins dans tous les cas. Or si on considère une image naturelle cet aspect est néfaste à proximité des contours car il favorisera la présence d'un dégradé. Aussi, de nombreuses fonctions ϕ ont été proposées pour modéliser les potentiels des cliques d'ordre 2 :

$$U_{c=(s,t)} = \phi(x_s - x_t)$$

En particulier, la fonction suivante permet de respecter les contours de l'image

$$\phi(x) = \frac{1}{1 + (\frac{x}{\delta})^2}$$

Et est donc très utilisée en restauration. Ces modèles permettent de synthétiser des textures très variées.

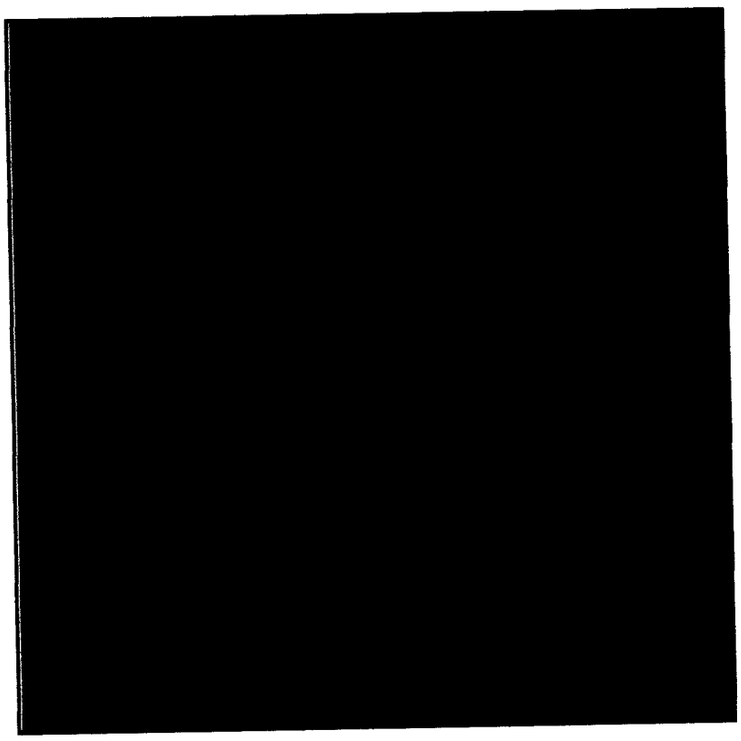


Figure 4.3 : Le modèle gaussien 4-connexe

$$U(x) = \sum_{c=(s,t)} (x_s - x_t)^2 + \alpha \sum_{s \in S} (x_s - \mu_s)^2$$

A	B
C	D

- **A** : $\alpha = 5.10^{-4}$
- **B** : $\alpha = 5.10^{-3}$
- **C** : $\alpha = 2.10^{-3}$
- **D** : $\alpha = \infty$ *($\mu = 127$) pour toutes simulations)*

[S16]

4.5. La modélisation bayésienne[6]

Soient:

y : l'observation
x : l'étiquette

On veut retrouver x à partir des informations contenues dans y

On cherche donc x sachant y :

$$\Pr(X = x \,/\, Y = y) = \frac{\Pr(Y = y \,/\, X = x) \,.\, \Pr(X = x)}{\Pr(Y = y)}$$ Loi de Bayes

$\Pr(X = x \,/\, Y = y) \propto$	$\Pr(Y = y \,/\, X = x)$.	$\Pr(X = x)$
↓	↓	↓
probabilité a posteriori de x	formation des observations	a priori

La probabilité de l'image sachant la scène (la formation de l'image). Souvent un modèle physique. Appelée la **vraisemblance.**

La probabilité de la scène avant d'avoir vu l'image. Appelée la probabilité **a priori.**

Les probabilités se simplifient quand quelques variables sont indépendantes les unes des autres.

Les champs de Markov sont une possibilité pour définir des probabilités simplifiées mais encore utiles.

4.6. Conclusion

Dans ce chapitre nous avons abordé le processus de segmentation basée sur les champs de Markov, dans ce contexte on a défini quelques notions de base (système de voisinage ...), et on a vu que l'utilisation du modèle Markovien dans le cadre de segmentation consiste à minimiser la fonction d'énergie.

On a remarqué aussi que l'un des domaines abordés lors de la segmentation et qui utilisent des modélisations stochastiques est la modélisation de textures ce qui est le but de notre étude.

Chapitre 5

Conception

5.1. Introduction

Après avoir terminer la première partie de notre projet qui a été consacré à l'étude théorique, nous commençons l'étude pratique. Cette dernière elle-même est décomposée en deux parties à savoir :

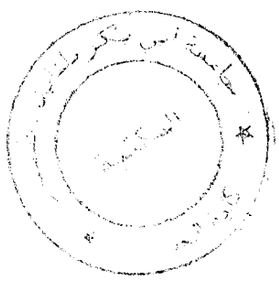
- La partie de conception (conception globale et conception détaillée).
- La partie de mise en œuvre (implémentation).

Le chapitre courant est orienté vers l'étude conceptuelle. Il est organisé de la façon suivante :

D'abord, nous donnons une description générale de notre système, en déterminant ses tâches principales.

En suite, nous allons construire sa conception globale en affecte à chaque module une tache bien définie, tout en restant ouvert à des communications externes alors nous abouti à un système modulaire avec haut degré de cohésion, et un faible couplage.

Enfin, nous achevons ce chapitre par une conception détaillée, dans laquelle nous allons détailler chaque module indépendamment, en précisons ses entrées, et la façon (Algorithme) qui permet de converge aux résultats voulus.



5.2. Description du système

L'objectif principal de notre système est de fournir des images segmentées à partir des images texturées acquises, autrement dit, un apprentissage de textures et pour cela en utilisant l'approche MRF comme illustré dans la figure suivante :

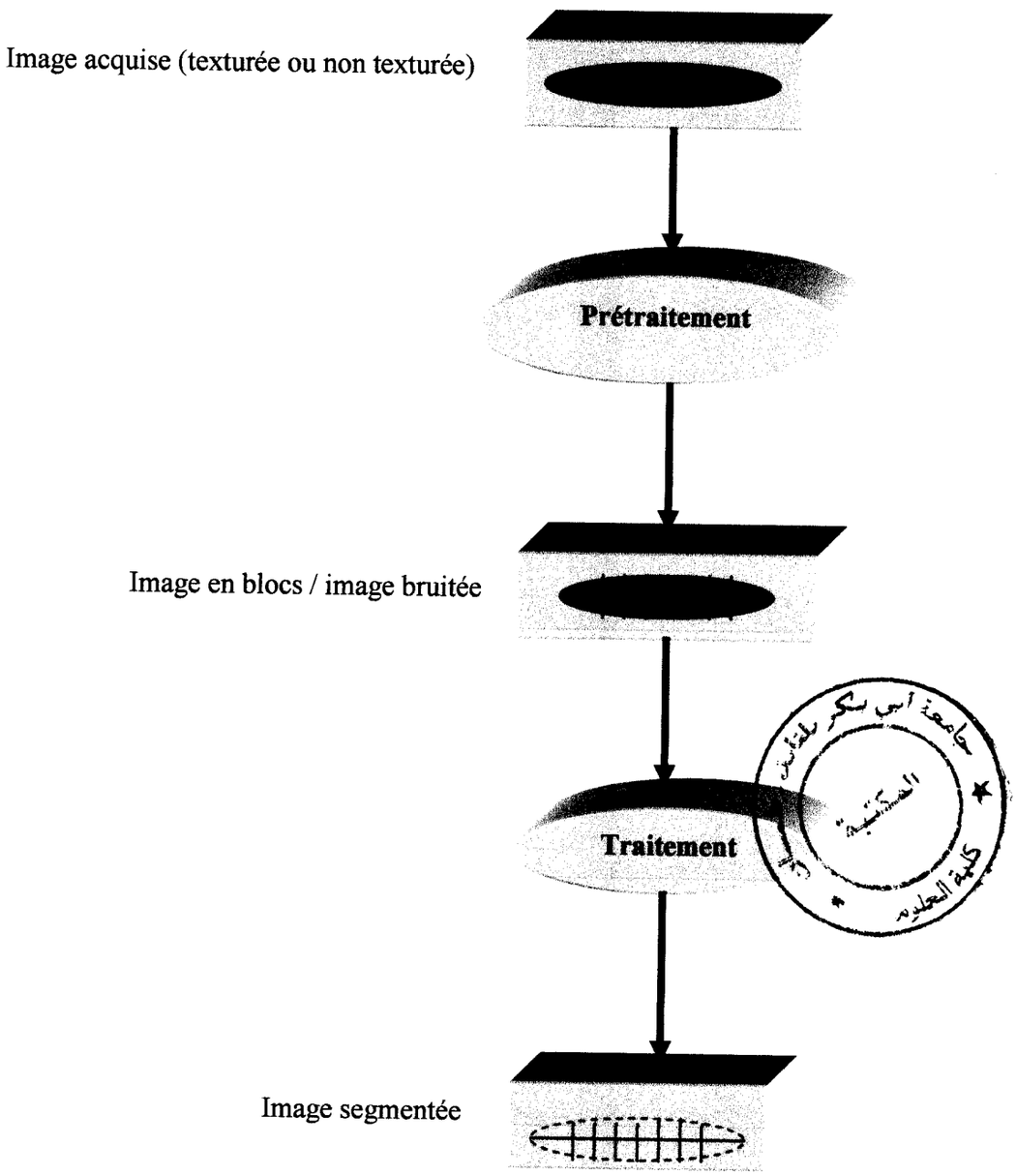


Figure 5.1 : Schéma général du système.

L'objectif de notre système c'est de réaliser les tâches suivantes :

- Découper les images texturées en un ensemble de blocs.
- L'initialisation (segmentation initiale des images texturées ou non texturées) par l'algorithme k-moyens adaptatif.
- La segmentation des images texturées on utilisant *les M.R.F.s.*

5.3. Conception globale

Notre système est conçu de tel sort qu'il assure un haut degré de cohésion, et un faible couplage, il se compose d'un ensemble de modules ou chaque module est spécialisé dans un travail ouvert à une communication avec les autres.

5.4. L'architecture globale du système

L'architecture globale de notre système est présentée dans la figure suivante :

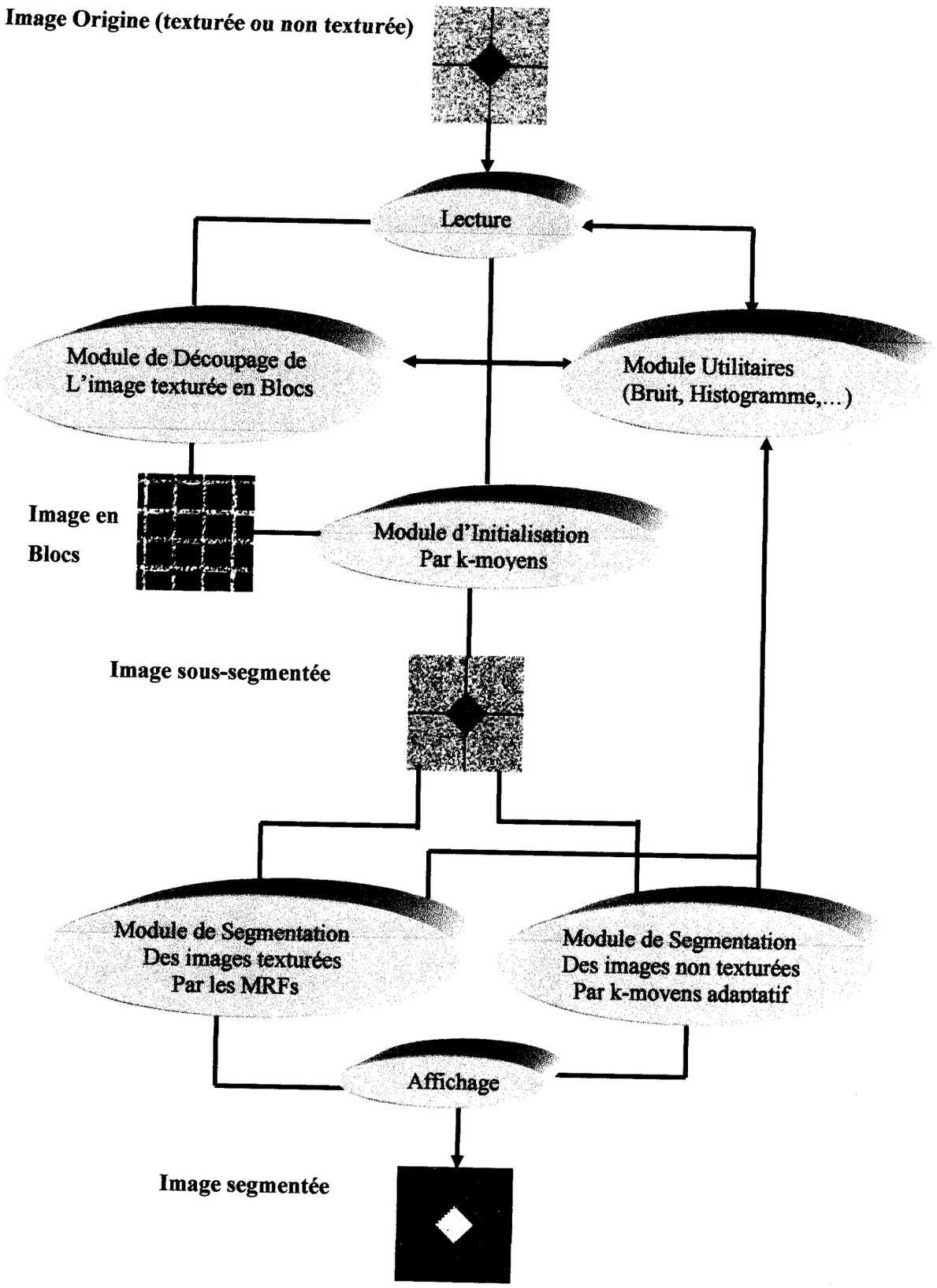


Figure 5.2 : l'architecture globale du système.

5.4.1 Image Origine (texturée ou non texturée)

L'image de traitement en format BMP (*Bit Map*), ce format est très utilisé en imagerie satellitaire, aérienne, médicale et images texturées.

5.4.2 Fonction Lecture d'Image

La lecture se fait à partir de fichier (fich.bmp), elle transforme l'image sous forme matricielle pour faire le traitement.

5.4.3 Fonction Ecriture

Après les traitements faites sur la matrice de l'image on peut afficher l'image segmentée, elle transforme un tableau à deux dimensions (résultat de traitement) vers une image (format bmp).

5.4.4 Fonctions utilitaires

Ce module contient toutes les fonctions utilitaires de notre système : fonction conversion des images couleur en des images niveau de gris, fonction de bruitage, histogramme, courbe d'énergie... etc.

- La conversion d'image couleur en image niveau de gris

Ce module a pour rôle de convertir l'image couleur en une image au niveau de gris.

- Histogramme

Il définie la variation entre les pixels d'image et les classes des niveaux de gris.

- Courbe

Elle définie la variation entre l'énergie globale et le des itérations.

5.4.5 Module pour découper l'image texturée

Ce module permet de découper l'image originale (texturée) en un ensemble de blocs, ces derniers sont généralement de taille fixe(8×8 par exemple).

Figure 5.3 : image originale (texturée)

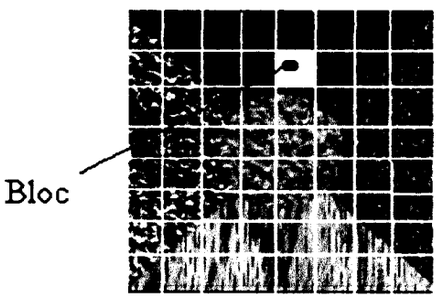


Figure 5.4 : image originale découpée en blocs.

5.4.6 Module d'Initialisation

L'intervalle de niveau de gris est divisé en N classes (selon le nombre de classes), et chaque bloc possède une classe quelconque voir fig. 5.5.

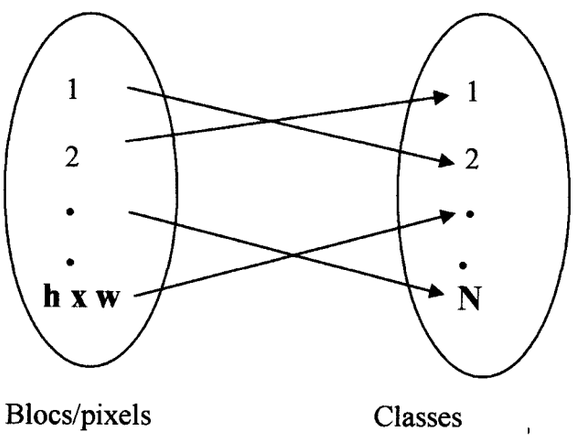


Figure 5.5 : La relation entre les blocs / pixels et les classes.

5.4.7 Module de segmentation des images par les M.R.F.s

Le rôle de ce module est de trouver l'image segmentée, pour cela, il affecte une valeur estimée pour chaque bloc (cas des images texturées) ou pour chacun des pixels (cas des images non texturées), à partir de la valeur de ses voisins.

5.5. Conception détaillée

Cette phase est orientée vers l'étude détaillée de chaque module et fonction, en précisant ses entrées, et l'algorithme adéquat pour arriver aux résultats voulus.

- Fonction lecture

Cette fonction doit fournir d'une part les services de la lecture nécessaire pour que le traitement soit simple, d'autre part il doit se présenter comme un outil de transformation (conversion d'image en un tableau de deux dimensions).

- Module utilitaire

Ce module à pour objectif de fournir plusieurs services (courbe d'énergie, Histogramme...etc.).

- Courbe : (*Energie =F(configuration)*)

Cette courbe définie la variation entre l'énergie globale et les configurations (image résultat). Il se fait à l'aide d'une sauvegarde de toutes les énergies globales pour chaque itération dans un tableau, et finalement dessiner la courbe.

- Histogramme : (*Nombre d'occurrence = F(niveau de gris)*

Cet histogramme défini la variation entre le nombre d'occurrence et le niveau de gris par parcours total de l'image et calcule de nombre d'occurrence de chaque niveau de gris, et le résultat est sauvegardé dans un tableau. Ces étapes sont présentées dans l'algorithme suivant :

Algorithme : La table d'occurrence

Initialisation *(la table d'occurrence est vide)*
Pour i *= 0 jusqu'à la longueur de l'image* ***Faire***
Pour j *= 0 jusqu'à la largeur de l'image* ***Faire***
Pour k *= 0 jusqu'à 255* ***Faire***
Si *le niveau de gris de site (i,j) =Table d'occurrence[k]* ***Alors***
Table d'occurrence[k]= Table d'occurrence[k] + 1
Fin pour
Fin pour
Fin pour

- Bruit gaussien

C'est une fonction basée sur la distribution de gauss, pour attribuer à l'image une suite de valeurs (bruits), pour faire cette opération il suffit de construire une image suivant la loi de gauss qui est défini comme suit :

- On calcule la moyenne de l'image cible, basée sur la formule mathématique suivante :

La moyenne = la somme de tous les niveaux de gris de l'image à traiter divisé par le nombre total de pixels.

L'écart type = la somme de (niveau de gris de chaque pixel – la moyenne) divisé par le nombre total de pixel.

- Module de segmentation des images

L'objectif de ce module est de trouver à partir d'une image texturée une image segmentée. Ce traitement est effectué par une modélisation *Markovienne* associée à l'estimation *Bayesienne*, cela revient à optimiser une fonction d'énergie globale.

$$U(\lambda/a) = U_1(\lambda/a) + U_2(\lambda)$$

Avec l'utilisation de l'estimateur MAP, le champ des étiquettes ŷ qui maximise la probabilité a posteriori **P(*y*/a)** est donné par :

$$\hat{y} = \max\,(p(y/a))$$

Par l'utilisation du théorème de Bayes :

$$P(\lambda/a)=\frac{P(a/\lambda)P(\lambda)}{P(a)}$$

Le champ des étiquettes $\hat{y}$ représente la segmentation voulue, qui est réalisée par les deux étapes suivantes :

- *La* première *étape* est la modélisation.
- *La deuxième étape est l'optimisation.*

- Modélisation

Pour la modélisation de p(λ), un modèle Markovien nous permettra de prendre en considération la dépendance spatiale entre les sites adjacents, l'image du champ des étiquettes (classes) est considérée donc comme une réalisation d'un champ aléatoire de Markov, ce dernier tien en compte la structure géométrique entre les pixels voisins, tel que la probabilité conditionnelle d'un pixel $\mathbf{s_1}$ dépend uniquement des probabilités de ses voisins (propriété des champs de Markov). Ceci s'exprime par la formule suivante :

$$P(\lambda_{s_1}/\lambda_{s_2}, s_2 \in s-\{s_1\})=P(\lambda_{s_2}, s_2 \in v_{s_1})$$

Où $\mathbf{v_{s1}}$: est le voisinage du site **s**.

D'après le théorème de ***Hammersly-Cliffortd,*** le champ aléatoire λ suit la loi de probabilité de *Gibbs* donnée par la formule suivante :

$$P(\lambda)=\frac{\exp(-U(\lambda))}{Z}$$

Où : ***Z*** : est une constante de normalisation.

U : est la fonction d'énergie, elle définie par la formule suivante :

$$U(a)=\sum_{c\in C} v_c(a/c)$$

Dans notre étude nous avons utilisé :

- Un système de voisinage d'ordre deux.
- Une fonction potentielle $\mathbf{V_2}$:

$$\mathbf{V_2}(\lambda)=\beta_1\, t_1\,(x)+\beta_2\, t_2\,(x)$$

$t1(x)$: Nombre de voisins différents horizontalement et verticalement.

$t2(x)$: Nombre de voisins différents diagonalement.

K : constante pour toute image. $\beta_1=\sqrt{2}\beta_2=K\lambda$.

- Une fonction d'énergie globale **U (λ , a)** :

$$U(\lambda,a)=\sum_{s\in S}\left(\ln\left(\sqrt{2\pi\sigma\lambda_s}\right)+\frac{(\lambda_s-\mu_\lambda)^2}{2\sigma_{\lambda_s}^2}\right)+\sum_{c\in C}V(\lambda)$$

- Une fonction d'énergie locale (selon Bouman):

- $$U_s=\frac{1}{2}\left(\frac{\tilde{y}}{\sigma_{x_s}}+\log(\sigma_{x_s}^2)+\log(2\pi)\right)+\sum_{c\in C}V(\lambda)$$

- $$U_s=\ln\left(\sqrt{2\pi\sigma\lambda_s}\right)+\frac{(\lambda_s-\mu_\lambda)^2}{2\sigma_{\lambda_s}^2}+\sum_{c\in C}V(\lambda)$$

L'estimation des paramètres du modèle est l'un des difficultés principales des méthodes Markoviennes, cette difficulté réside dans l'estimation du paramètre $\tilde{y}$ de l'expression de la fonction d'énergie selon Bouman, ou $\tilde{y}=y_s-\hat{y}$, donc comment estimer le paramètre ŷ ?

$$\hat{y}_{m,n}=-\sum(i,j)\in D_{p,r}^h a_{i,j}y_{m-i,n+h-j}$$

- Optimisation

L'objectif de cette étape est de trouver une configuration λ d'étiquettes correspond à une configuration initiale ***a*** de l'image observée à condition quelle maximise la distribution de probabilité a posteriori **P (λ/a)** et minimise la fonction d'énergie **U (λ/a) .**

Deux types d'algorithmes sont utilisés pour faire cette optimisation à savoir :

- Les algorithmes *aléatoires (Stochastiques).*
- Les algorithmes *déterministes.*

Dans notre étude nous allons implémenter l'algorithme déterministe I.C.M (Iterated conditional modes).

L'algorithme I.C.M est présenté de la façon suivante :

Initialisation.

Répéter

Pour i = 0 jusqu'à la longueur de la matrice ***Faire***

Pour j = 0 jusqu'à la largeur de la matrice ***Faire***

Extraire la matrice de voisinage.

Pour q = 0 jusqu'au nombre des classes ***Faire***

- Affecter au bloc(i,j) le niveau de gris de la classe q.
- Calculer la fonction d'énergie de bloc(i,j) à partir de la matrice de voisinage.
- Sauvegarder l'énergie de bloc(i,j) dans une matrice.

Fin pour

Calculer le minimum des valeurs de la matrice des énergies.

Affecter le niveau de gris de la classe q qui correspond à la valeur d'énergie Minimale au bloc (i,j).

Fin pour

Fin pour

Jusqu'à une condition d'arrêt

Algorithme : Modes conditionnels itérés.

Initialisation *(K-moyen)*

Taille ←Taille de l'image ;

Ni ←1 ; (nombre d'itération)

Répéter

Répéter

Estimer la nouvelle $\hat{\mu}_s^l$; (fonction d'intensité)

Estimer $\hat{x}$; (champ des étiquettes)

Ni←Ni+1 ;

Jusqu'à ($\hat{x}$ converge Ou Ni=Nimax)

W←W/2 ;

Ni←1 ;

Jusqu'à (W=Wmin)

K-moyen adaptatif

Chapitre 6

Implémentation & résultats obtenus

6.1. Introduction

Il existe plusieurs approches qui traitent la segmentation par région, elles visent à l'extraction des indices visuels. Généralement, ces indices sont des régions de l'image.

Les méthodes appartenant à cette famille manipulent directement des régions. Soit, elles partent d'une première partition de l'image, qui est ensuite modifiée en divisant ou regroupant des régions.

Nous présentons dans ce chapitre en détail les résultats obtenus de l'opération de segmentation, en expliquant, pas à pas, comment s'est réalisée l'implémentation ainsi que les outils qui ont servi à la réalisation de notre logiciel.

6.2. Choix du langage de programmation [S11]

Notre application est réalisée en utilisant le compilateur Borland C++ Builder version 6.

C++ Builder est un environnement de développement proposé par Borland fondé sur le C++. Borland a repris les recettes développées avec succès pour ses produits phares orienté Pascal : Delphi ; notamment, l'interface et les bibliothèques de composants. La version 6 a vu le jour en 2002 et après de nombreuses mises à jour, a été remplacée par une nouvelle version baptisée Borland C++ Builder 2006.
C++ Builder est un outil RAD (Rapid *Application Development)*, c'est-à-dire tourné vers le développement rapide d'applications sous Windows. C++ Builder permet de réaliser de façon très simple l'interface des applications et de relier aisément le code utilisateur aux événements Windows, quelle que soit leur origine (souris, clavier, système).

Pour ce faire, C++ Builder repose sur un ensemble très complet de composants (visuels et non visuels) prêts à l'emploi. La quasi-totalité des contrôles Windows (boutons, boîtes de saisie, listes déroulantes, menus, ...) y est représentée. Les caractéristiques de ces composants sont éditables directement dans une fenêtre spéciale intitulée **inspecteur d'objets**. Un clic de souris sur les composants permet d'associer du code à l'objet.

6.2.1. L'environnement de développement C++ Builder

Nous allons commencer la prise en main du produit en décrivant l'environnement de développement.

L'interface

La figure suivante représente un exemple typique de l'interface au cours d'une session de travail.

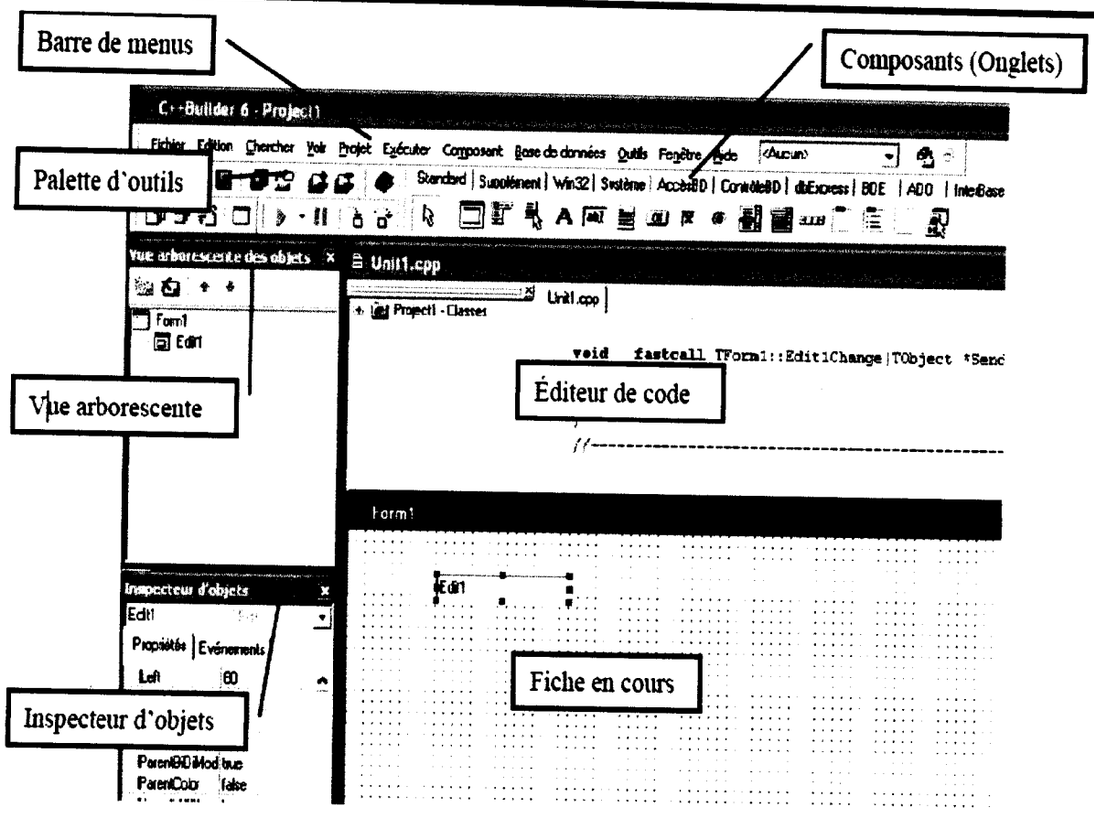


Figure 6.1 : l'interface graphique de C++ builder

L'interface peut paraître déroutante car elle composée de différentes fenêtres qui ne recouvrent pas tout l'écran. Les applications qui ont été préalablement lancées sont toujours visibles.

On peut distinguer plusieurs zones distinctes :

- La barre de menu.
- La barre d'outils qui se décompose elle-même en deux parties :
  - La palette d'outils pour les opérations classiques.
  - La palette de composants rangés par catégories (onglets).
- Une fiche ou Forme en Anglais qui représente l'interface en cours de création.

Si l'application comporte plusieurs fiches, elles sont cachées et disponibles par le menu et le raccourci clavier F12. Du fait de l'environnement RAD, elles représentent à l'identique les composants à l'exécution (en dehors des composants non visuels).

L'inspecteur d'objets qui donne les caractéristiques de l'objet sélectionné dans la fiche, tant au niveau des propriétés (attributs) que des événements.

L'éditeur de code avec affichage automatique du code lié à l'objet sélectionné dans la fiche.

A chaque fiche correspond deux fichiers : un fichier en-tête (.h) et un fichier code (.cpp) éditables.

D'autres fenêtres suivant le contexte peuvent être disponibles : arborescence des objets, débogueurs, experts.

62.2. Les composants de C++ Builder

Par défaut, C++ Builder utilise un compilateur C++, un éditeur de liens, un compilateur de ressources et un gestionnaire de projets intégrés.

Il est toutefois possible de spécifier la volonté d'utiliser les outils en ligne de commande livrés avec C++ Builder ou bien d'autres outils tiers. Cette dernière possibilité est très utile lorsque l'on veut utiliser des modules compilés avec d'autres langages.

6.3. Implémentation

Nous présentons dans cette partie les différentes interfaces de notre application, les différents résultats du traitement, ainsi l'interprétation de ces résultats.

6.3.1. Environnement matériel et logiciel de programmation

Notre application a été réalisé sur un PC de type Pentium core 2 duo 1.6 GHz : 1Go de RAM sous Windows XP.

6.3.2. Présentation de quelques vues

L'application est composé d'une fenêtre principale elle contient un menu déroulant permettant d'accéder aux différentes fonctionnalités voire figure suivante :

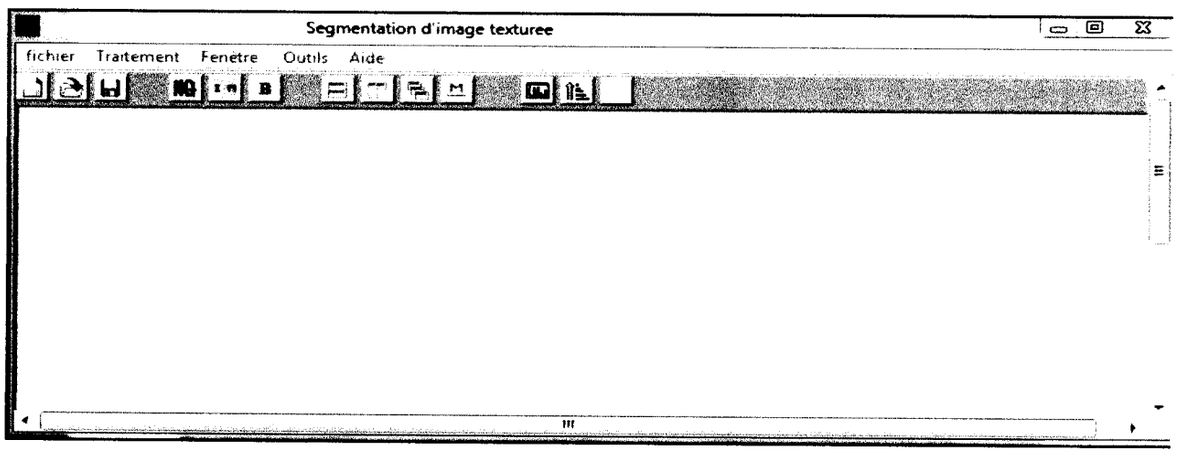


Figure.6.2 : Fenêtre principale

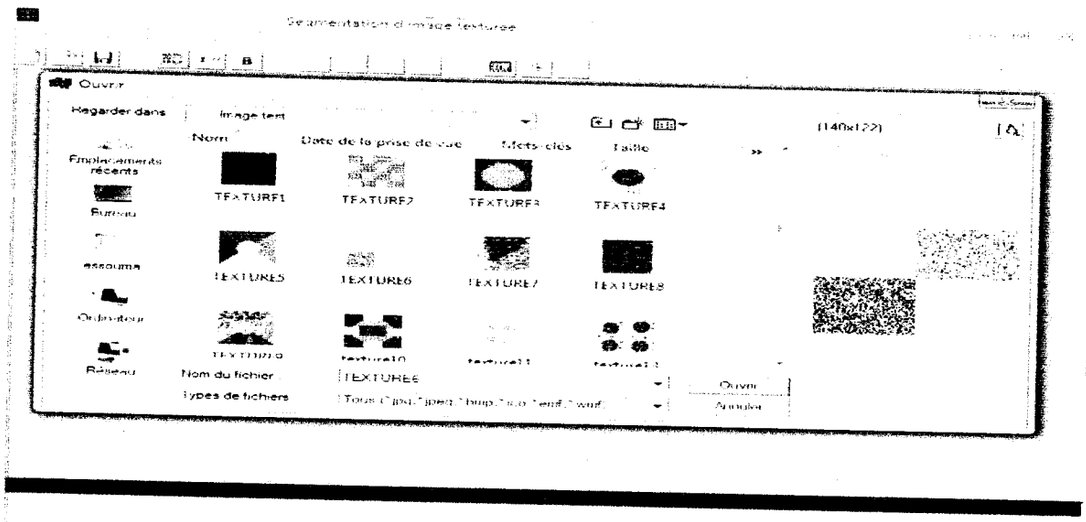


Figure 6.3 : Fenêtre d'affichage

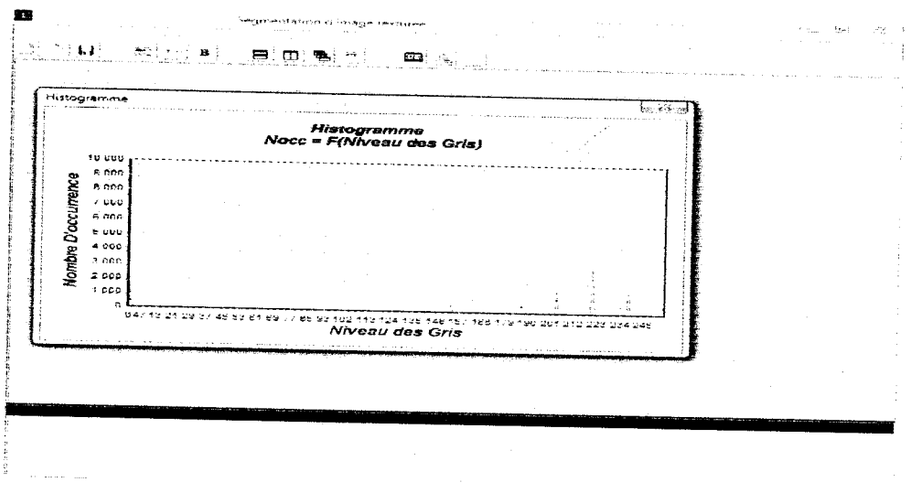


Figure 6.4 : Fenêtre d'histogramme

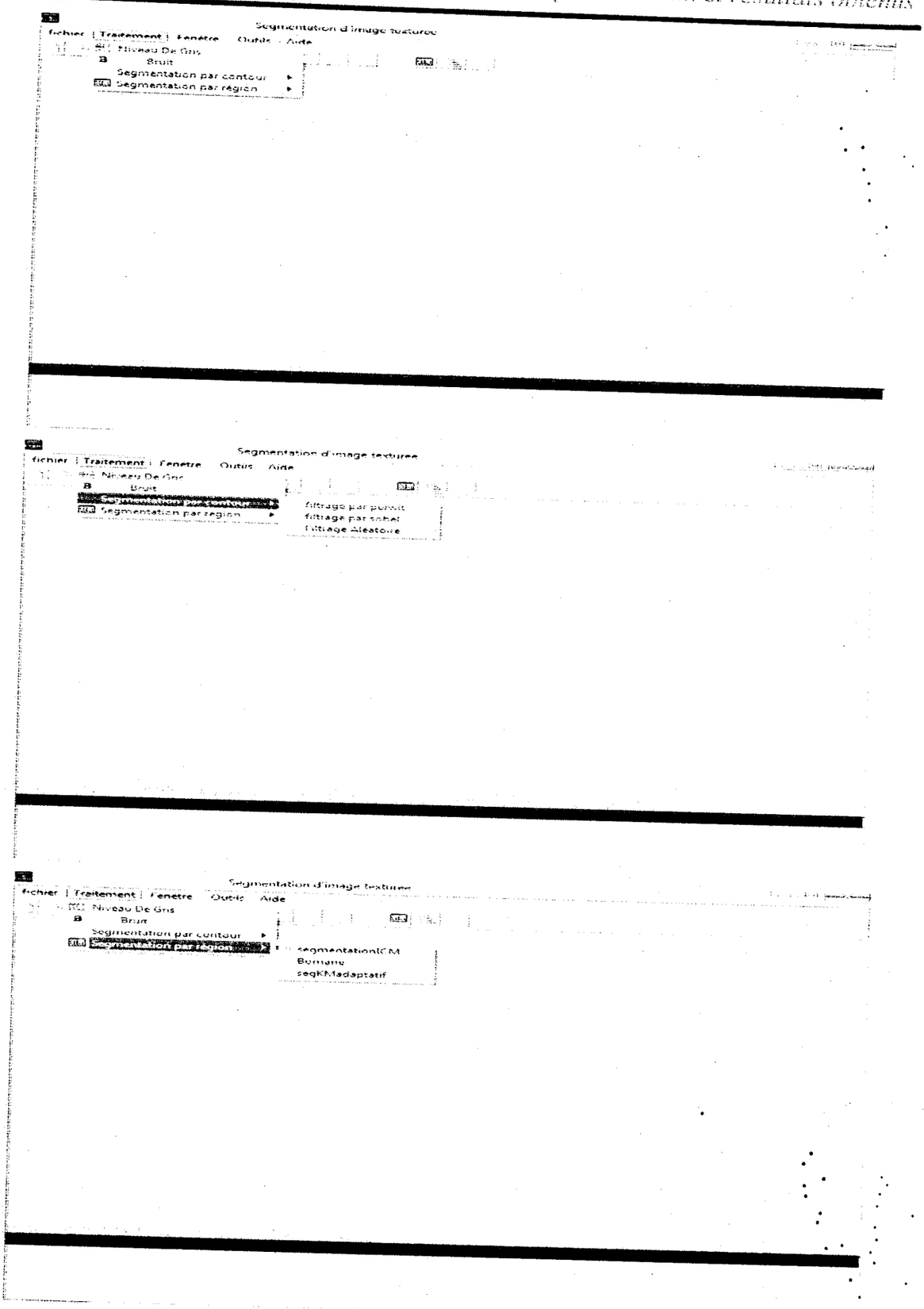


Figure 6.5: Fenêtre représentant les méthodes de traitements

6.4. Résultats expérimentaux

6.4.1. L'algorithme de Sobel

C'est un opérateur utilisé en traitement d'image pour la détection de contours. Il s'agit d'un des opérateurs les plus simples qui donne toutefois des résultats corrects.

Description simplifiée

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

Dans la figure qui suit nous avons appliqué Sobel pour la détection de contour

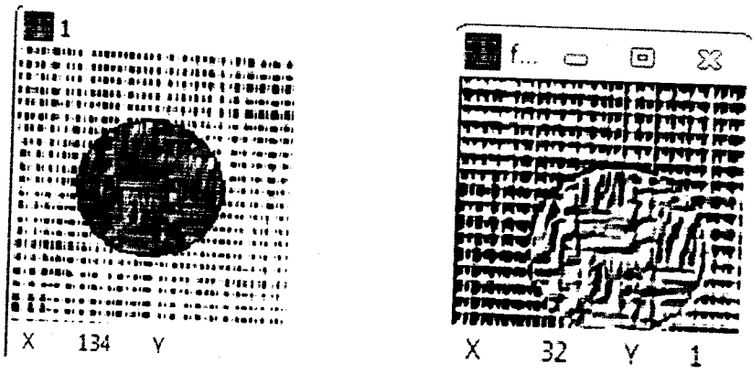


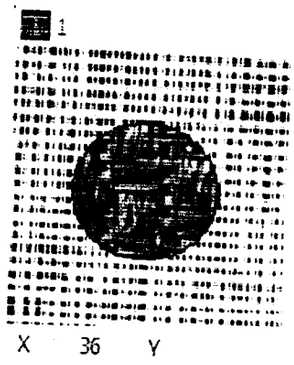
Image Original **Contour détecter par sobel**

Figure 6.6 : représentation du filtre Sobel

6.4.2. Filtre de Prewitt

La matrice qui correspond au filtrage horizontal, faisant ressortir essentiellement les contours verticaux, selon l'opérateur de Prewitt, s'écrit $h_x = [-1\ 0\ 1]$ tandis que la matrice verticale h_y est sa transposée. Les deux convolutions avec le tableau de valeurs initiales créent deux tableaux G_x et G_y à l'origine du tableau G sur lequel on peut localiser les maximums.

Application de Perwit pour la détection de contour



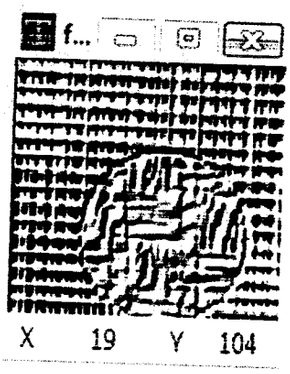


Image Originale **Contour détecter par perwit**

Figure 6.7 : représentation du filtre Perwit

6.4.3. Test de segmentation des images par MRF

Test 1

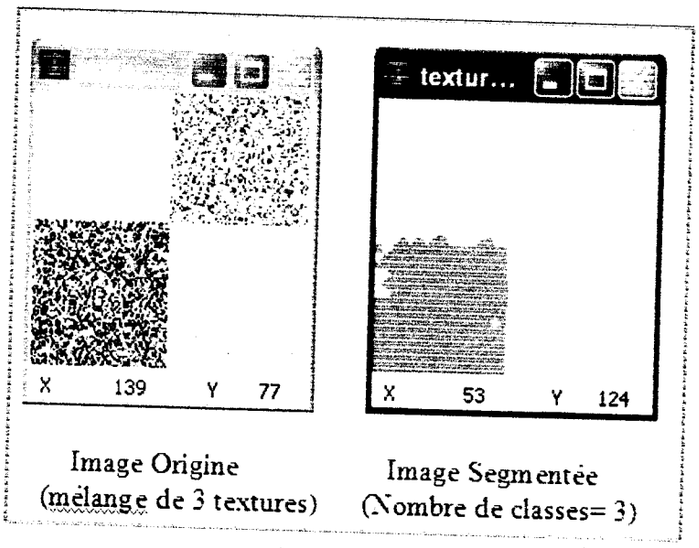


Image Origine (mélange de 3 textures)

Image Segmentée (Nombre de classes= 3)

Test 2

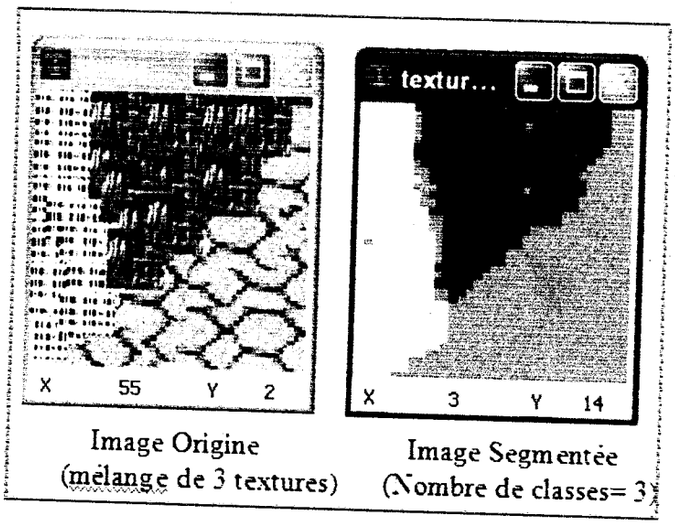


Image Origine (mélange de 3 textures)

Image Segmentée (Nombre de classes= 3)

Test 3

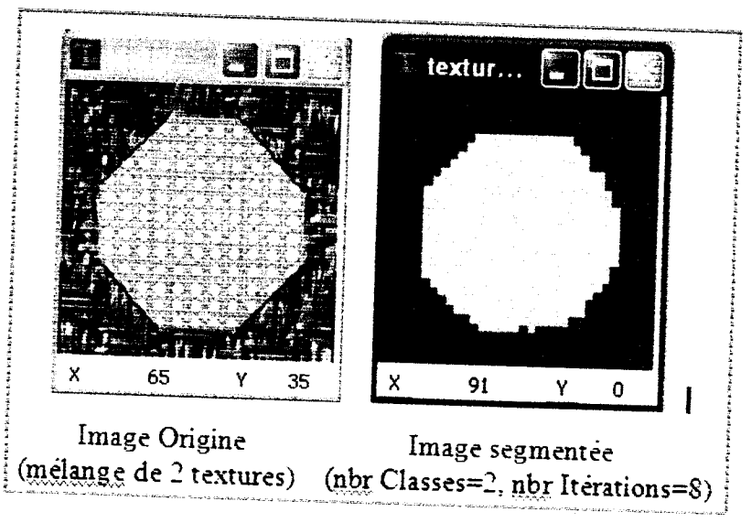
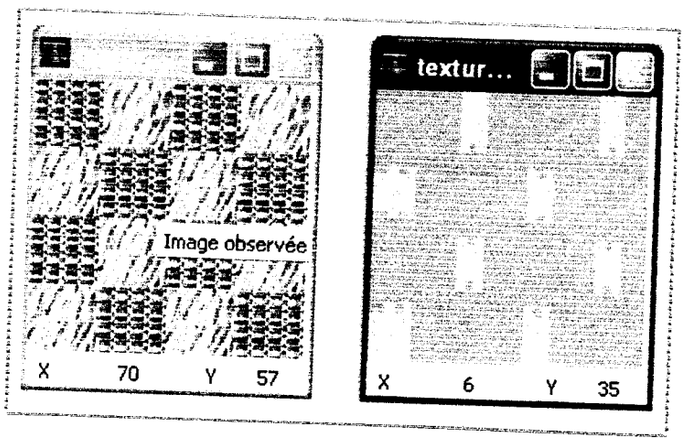


Image Origine (mélange de 2 textures)

Image segmentée (nbr Classes=2, nbr Itérations=8)

Test 4



Dans le test 4, il est clair que le résultat de segmentation de l'image origine (mélange de deux textures) est moyen car le niveau de gris des deux textures est très proche.

Test 5

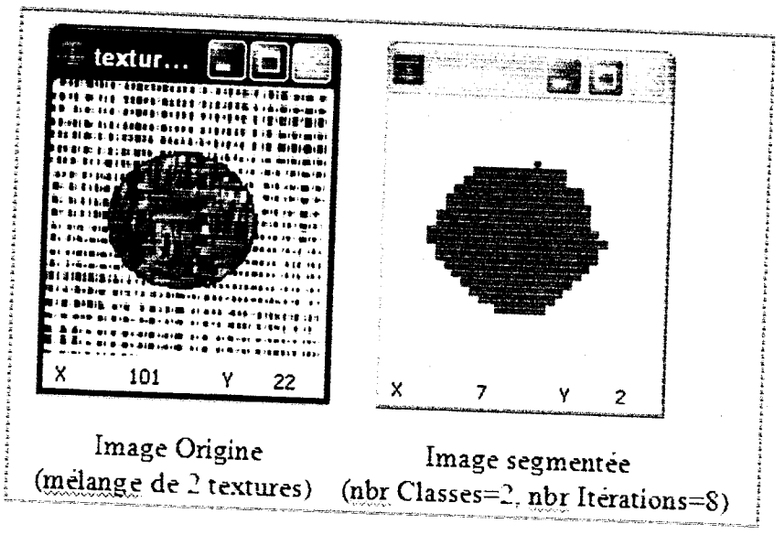


Image Origine (mélange de 2 textures)

Image segmentée (nbr Classes=2, nbr Itérations=8)

Test 6

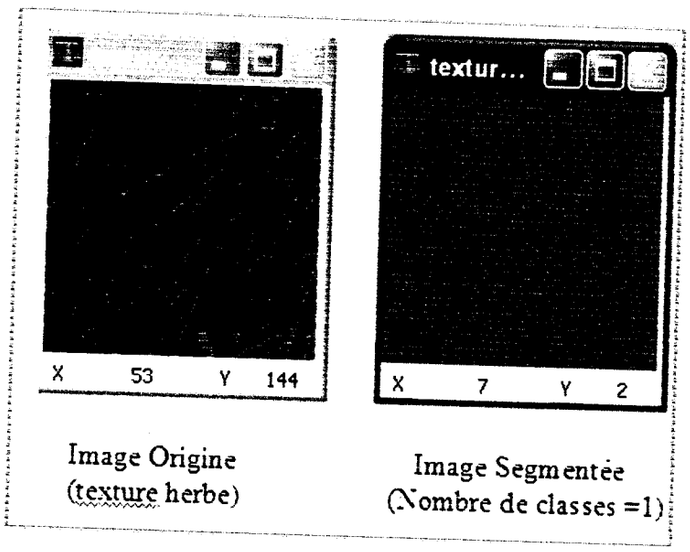


Image Origine (texture herbe)

Image Segmentée (Nombre de classes =1)

6.4.4. ICM (iterated conditional modèle)

L'algorithme ICM est un algorithme déterministe car chaque pixel est a été systématiquement à la classe k qui rend l'énergie U=U1+U2 minimale. Les grandes étapes de l'algorithme sont les suivantes :

Initialisation : L'algorithme ICM est initialisé à l'aide de l'image classique existante. Dans notre cas, elle est issue d'une classification utilisant l'algorithme des K-moyen.

Balayage de l'image : En chacun des pixels, on calcule l'énergie U relativement a chaque classe qui rend son énergie minimale.

Test d'arrêt de l'itération : Si il y'a un changements de classes lors de l'étape 2 ou si le nombre d'itérations donnée n'est pas atteint c'est a dire 100 dans notre cas, on recommence l'étape précédente, sinon on arrête l'itération.

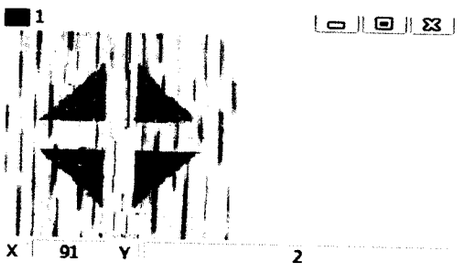


Image original

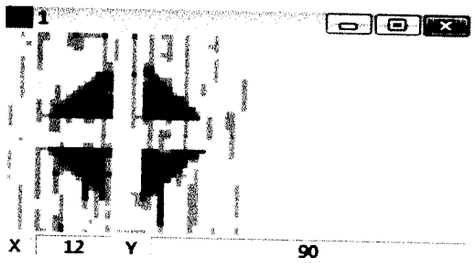


Image Segmentée en 3 classes

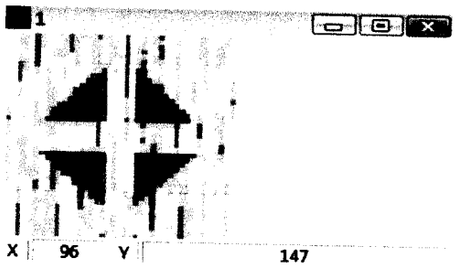


Image Segmentée en 4 classes

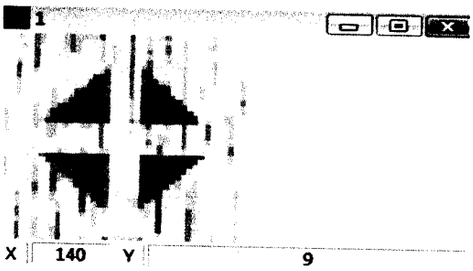


Image Segmentée en 6 classes

Figure 6.8 : Résultats de segmentation par ICM

6.4.5. BOMANE

L'algorithme de Baumane

Est un algorithme utilisé pour estimer les paramètres d'un modèle de Markov.

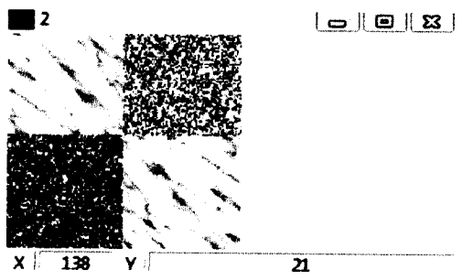


Image original

Image Segmentée en 3 classes

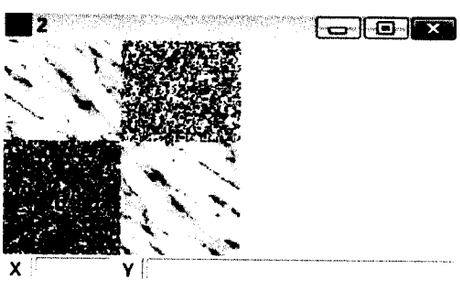


Image Segmentée en 6 classes

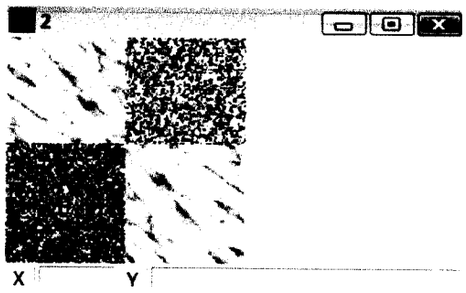


Image Segmentée en 9 classes

Figure 6.9 : résultats de segmentation par Bomane

6.4.6. K-MOYEN ADAPTATIF

K-means

Une des techniques de classification non supervisée des plus utilisées. Etant donné un entier K, K-means partitionne les données en K groupes, ou "classes" ne se chevauchant pas. Ce résultat est obtenu en positionnant K "prototypes", dans les régions de l'espace les plus peuplées. Chaque observation est alors affectée au prototype le plus proche (règle dite "de la Distance Minimale"). Chaque classe contient donc les observations qui sont plus proches d'un certain prototype que de tout autre prototype. Les prototypes sont positionnés par une procédure itérative qui les amène progressivement dans leur position finale stable.

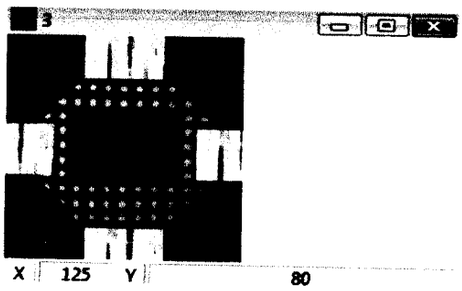


Image original

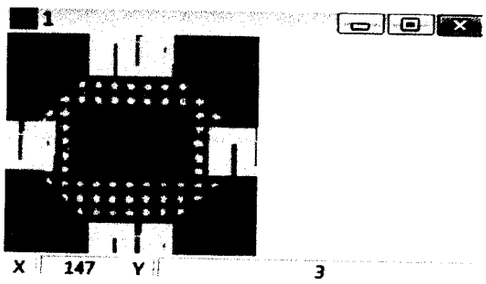


Image Segmentée en 4 classes

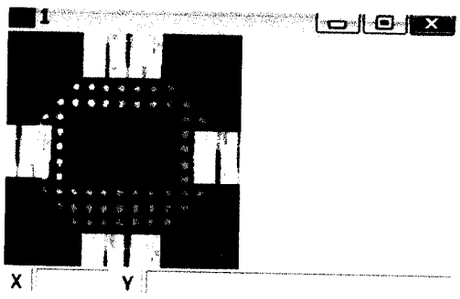


Image Segmentée en 8 classes

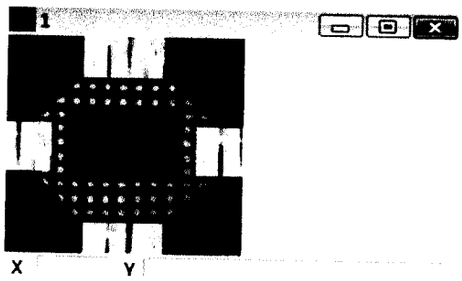


Image Segmentée en 12 classes

Figure 6.10 : Résultats de segmentation par K-moyennes

6.5. Code source de notre application

Dans notre application une image "I" peut être représentée à travers une matrice de N*M éléments. Où N est le nombre de lignes « longueur de l'image » et M est le nombre de colonnes « largeur de l'image ».

Pour la raison de grande taille de ces matrices, typiquement liées à la taille l'image, nous préférons utiliser l'allocation dynamique de l'espace mémoire associé à ces matrices pour qu'on puisse les supprimer facilement "lorsque la suppression est nécessaire ". Alors I a la structure suivante :
typedef int PIXEL ;

PIXEL **S =new PIXEL*[N];

for(int i = 0 ; i < N ; i ++)

S[i]= new PIXEL **[M];**

Donc S est un pointeur d'une matrice de nombres entiers représentant le niveau de gris de chaque pixel, dont on alloue pour chaque image, seulement la taille utile et nécessaire.

Exemple

Nous citons quelques matrices utilisées dans notre application :

Mvoisin : est une matrice qui contient les valeurs de niveau de gris des blocs voisins à un bloc donné.

Morigine : est une matrice qui contient les valeurs de niveau de gris des pixels de l'image origine.

Mmoy : est une matrice qui contient les valeurs de niveau de gris des blocs " après que l'image est découper en blocs".

On a utilisé l'allocation dynamique aussi pour faire l'allocation des tableaux. Pour un tableau Tab qui contient des valeurs entières la structure est comme suit :

Int *Tab = new int [N];

Exemple

Nous citons quelques tableaux utilisés dans notre application :

TNG : est un tableau qui contient les niveaux de gris moyens des classes de l'image.

TFE : est un tableau qui contient les valeurs de fonctions d'énergie.

```
Tfp *fp;

typedef int PIXEL;

float vc,TFE[20];

int nc,h,w,h1,w1; //- Déclaration de quelques variables glogales -

int NG[50];

AnsiString CurrentFile;

Graphics::TBitmap* Im1;

//----------------------- Possibilité de créer(ouvrir) plusieurs images ----------------------------

void __fastcall Tfp::CreateImage(String Name)

{

  Tf1* fils;

  fils=new Tf1(Application);
```

```
    fils->Image1->Picture->LoadFromFile(Name);

    fils->Caption=ExtractFileName(Name);

}

//---------- initialiser la matrice de voisinage par la valeur(-1) ----------

void Tfp::initial(PIXEL **Mvoisin)

{

int i,j;  //- initialiser la matrice de voisinage par la valeur(-1)

 for (i=0;i<3 ;i++)

  for (j=0;j<3 ;j++)

   Mvoisin[i][j]=(-1);

}

//---------- Calcule de moyen dans le système de voisinage----------


int Tfp::CalculMoyen(PIXEL **Mvoisin)
{
 int i,j,n=0;
 float som=0;
 for (i=0;i<3 ;i++)
  for (j=0;j<3 ;j++)
  {
   if(Mvoisin[i][j]!=(-1)){
                    som=som+Mvoisin[i][j];
                    n++;
                   }
  }
  som=som/n;
  return(som);
}
//---------- Calcule de la variance (système de voisinage 8-connexité) ----------
```

```
int Tfp::CalculVariance(PIXEL **Mvoisin,int U)
{
int i,j,som=0,p,n=0;
 for (i=0;i<3 ;i++)
  for (j=0;j<3 ;j++)
  {
   if(Mvoisin[i][j]!=(-1)){p=Mvoisin[i][j]-U;
                    som=som+(p*p);
                   n++;
                  }
  }
 som=som/n;
 return(som);
}
//------------------------- Calcule de la fonction d'énergie ----------------------------------
float Tfp::foctENER(PIXEL **Mvoisin,PIXEL **Mmoy)
 {
int jj,i,j,c,u,v;
 float s=0,o,vfe,f;
 u=CalculMoyen(Mvoisin);
 v=CalculVariance(Mvoisin,u);
 c=sqrt(6.28*v);
  for(i=0;i<h1;i++)
   for(j=0;j<w1;j++)
     {
     o=(Mmoy[i][j]-u);
     o=(o*o);
     f=o/(2*v);
     s=s+log(c+f);
     }
 vfe=s+vc;
 return(vfe);
}
//--------------------------- fonction potentiel selon Bouman et Liu ----------------------------
float Tfp::fonctPOT(PIXEL **Mvoisin)
```

```
{
 int l,e,K,L,s1=0,s2=0;
 float S=0;
   for (l=0;l<3;l++)   //for 3
    {
     L=1-l;
     for (e=0;e<3;e++)  //for 4
      {
       K=1-e;
       if ((Mvoisin[l][e]>= 0)&(Mvoisin[l][e]!=Mvoisin[1][1]))    //if 2
        {
  if (((L==1)&(K==1))|((L==-1)&(K==-1))|((L==1)&(K==-1))|((L==-1)&(K==1))) s1++ ;
   if (((L==1)&(K==0))|((L==-1)&(K==0))|((L==0)&(K==1))|((L==0)&(K==-1)))  s2++ ;
        } //fin if2
      }//fin for4
    } //fin for3
 S=(0.621*s1)+(0.439*s2);
 return(S);
}
//------------------------------ le minimum de la fonction d'énergie ------------------------------
 int Tfp::minimumfe()
 {
  int ind=0;
  float min=TFE[0];
  for(int k=1;k<nc;k++)
   if(TFE[k]<min){min = TFE[k]; ind=k;}
    min=NG[ind];
   return(min);  //- la meilleur classe
 }
//---------------------------------l'image en Niveau de gris-----------------------------------
void Tfp::NGris()
{
int i,j;
byte R,V,B,mg;
```

```
Tf1* ptr;
 ptr=dynamic_cast<Tf1*> (ActiveMDIChild);
 Im1 = new Graphics::TBitmap();
 h=ptr->Image1->Picture->Bitmap->Height;
 w=ptr->Image1->Picture->Bitmap->Width;
 Im1->Height = h;
 Im1->Width = w;
      for (int i = 0; i < h; i++)
              for (int j = 0; j< w; j++)
              {
              R=ptr->Image1->Picture->Bitmap->Canvas->Pixels[j][i];
            V=ptr->Image1->Picture->Bitmap->Canvas->Pixels[j][i]>>8;
            B=ptr->Image1->Picture->Bitmap->Canvas->Pixels[j][i]>> 16;;
            mg=R+V+B/3;
                Im1->Canvas->Pixels[j][i]=RGB(mg,mg,mg);
          }
}
void segmentation _ICM()
{
PIXEL **Mvoisin = new PIXEL*[3]; for(int k=0;k<3;k++)Mvoisin[k]= new PIXEL[3];
inti(Morigine);
for (i=0;i<h1;i++)
  {
  k=i*4;
  for (j=0;j<w1;j++)
   {
    l=j*4;s=0; initial(Mvoisin);
  ExSysVoi(i,j,Mmoy,Mvoisin);
  vc=fonctPOT(Mvoisin);
   for(q=0;q<nc;q++)
     {
      Mvoisin[1][1]=NG[q];
      TFE[q]=foctENER(Mvoisin,Mmoy);
     }
    min=minimumfe();
```

```
        for (q=k;(q<k+4)&(q<h);q++) for (p=l;(p<l+4)&(p<w);p++)Im1->Canvas-
>Pixels[p][q]=RGB(min,min,min);
    }
   }
   for(int k=0;k<3;k++)
 {delete[] Mvoisin[k];}
 for(int k=0;k<h1;k++)
 { delete[] Mmoy[k]; }
 for(int k=0;k<h;k++)
 {delete[] Morigine[k];}
 Tf1* fils;
    fils=new Tf1(Application);
    fils->Image1->Picture->Bitmap->Assign(Im1);
    fils->Caption=ptr->Caption;
//ptr->Image1->Picture->Bitmap->Assign(Im1);
}
delete(Im1);
}
//-------------------Bruit Gaussien (Prétraitement)-------------------------

void __fastcall Tfp::BruitExecute(TObject *Sender)
{
f3->TrackBar1->Position=0;
f3->ShowModal();
if (f3->ModalResult==mrOk)
{
 int i,j,t,bruit;
  Tf1* ptr;
 ptr = dynamic_cast<Tf1*> (ActiveMDIChild);
 h=ptr->Image1->Picture->Bitmap->Height;
 w=ptr->Image1->Picture->Bitmap->Width;
bruit=f3->TrackBar1->Position;
if(bruit!=0)
 {
 int bruit = f3->TrackBar1->Position;
```

```
Graphics::TBitmap* Im1;
Im1 = new Graphics::TBitmap();
Im1->Height = h;
Im1->Width = w;
Im1->Assign(ptr->Image1->Picture->Bitmap);
 t=(h*w*bruit)/100;
 for (int ya=0;ya<t ;ya++)
  {
   i= rand() % h;
   j= rand() % w;
   Im1->Canvas->Pixels[j][i]=RGB(125,125,125);
  }
   ptr->Image1->Picture->Bitmap->Assign(Im1);
   delete(Im1);
}}}
```

6.6. Conclusion

Dans ce chapitre, nous avons présentés les différents tests et résultats obtenues de la segmentation des images texturées; nous avons aussi montré quelque représentation de la structure de données qui est importante pour l'implémentation notre implémentation.

Après les différents tests effectués nous avons conclu que :

Le nombre de classes doit être choisi de façon empirique par l'utilisateur. Le nombre de classes dépend grandement de la nature de l'image.

Les résultats obtenus après les expérimentations sur les images texturées par les trois algorithmes, nous avons constaté que le nombre de régions diminue tant que le nombre de classes accroît.

Conclusion Générale

Conclusion générale

Dans notre travail, nous avons abordé le problème de l'analyse des images texturées.

Tout au long de notre étude, nous avons constaté que l'analyse d'image est un domaine très vaste, on peut le partager en deux étapes, la première, correspond aux différents traitements effectués sur les images pour les améliorer et les préparer à des traitements automatiques. La deuxième étape est l'ensemble des traitements permettant l'interprétation des images afin de pouvoir identifier les objets espérés parmi les autres objets existants.

La phase de segmentation par région dans un système de traitement d'image permet de faciliter l'extraction des objets significatifs dans des images texturées. Dans notre mémoire, nous avons utilisés des algorithmes de segmentation qui consistent à regrouper en région les pixels qui vérifient le critère d'homogénéité.

Les résultats obtenus par cette étude sont :

1- Présentation de différentes méthodes de segmentation.

2- Implémentation des algorithmes de segmentation des images texturées basées sur le champ Markov (MRF).

La qualité de l'interprétation d'une image dépend fortement de celle de la segmentation, malgré la grande diversité des méthodes. Les résultats de segmentation restent moyens et varient beaucoup en fonction de la technique choisie. Une méthode générale et automatique est difficile à concevoir étant donnés, les différents types de régions pouvant être présentes dans une image. Les résultats obtenus ne sont pas une fin en soit, au contraire ils nous mènent à poser les questions sur d'autres perspectives intéressantes comme, une coopération régions-contours qui met en correspondance nos algorithmes.

Nous remarquons que ce qui s'ouvre en perspective est plus large de ce qui a été accompli puisque, l'analyse d'image a surpassé le but qu'elle s'était fixée au début ; ce, qui n'était d'autre que de remplacer l'observateur humain par la machine.

Bibliographies

Articles, Livres Et Thèses

[1]: Oliver Alata – Clarisse Ramananjarasoa "Unsupervised textured segmentation using 2-D quarter plane autoregressive model with four prediction supports ", Pattern Recognition Letters 2004.

[2]: D.Clayssen / D.Lobstein / J.ZEITOUN, " Les Nouvelles images, introduction à l'image informatique", Edition DUNOD 1999.

[3] :J.P.Coquerez, S.Philipe, " Analyse d'images : Filtrage et Segmentation", Edition Marson1995.

[4] : Encyclopédie Microsoft Encarta 2004.

[5] : Michel.Claude.Girard, "Traitement des données de télédétection", Edition DUNOD 1999.

[6]: Huawu Deng, David A.Clausi, " Unsupervised image segmentation using a simple MRF model with a new implementation scheme",
PATTERN RECOGNITION 2004.

[7]: JAEHYUN PARK, LUDWIK KURZ, "Unsupervised Segmentation of Textured Images", NORTH-HOLLAND 1996.

[8] : Olivier Alata, " Caractérisation de textures par coefficients de réflexion 2-D Application en classification et segmentation", Thèse de doctorat de L'université Bordeaux I, école de doctorale, sciences physiques de l'ingénieur, 09 janvier 1998.

[9]: segmentation d'image basée sur MRF, Thèse d'ingéniorat, Biskra 2002.

[10] : Segmentation d'images basée sur les MRFs par l'approche SMA, Thèse d'ingéniorat, Biskra 2004.

[11]: Yihua Yu, Qiansheng Cheng " MRF parmeter estimation by an accelerated method", Pattern Recognition Letters 2003.

Sites Internet

[S1] : http://www.gtr.intv.parus13.fr

[S2] : http://www.inrialpes.fr

[S3] : http://www.kaddour.com

[S4] : http://www.ccr.jussieu.fr

[S5] : http://www.physpc21.u-strasbg.fr

[S6] : http://www.peros.club-internet.fr

[S7] : http://www.jedia.afia-France.org

[S8] : http://www.esil.univ-mrs.fr

[S9] : http://www.robots.ox.ac.uk.

[S10] : http://www.inria.fr

[S11]: http://www.apmep.asso.fr

[S12]: http://www.tsi.enst.fr

[S13] : http://www.gdr-isis.org

[S14] : http://www.unv-ls.fr

[S15]: http://www.irit.fr

[S16]: http://www.u-bordeaux1.fr

[S17]: http://www.univ-lr.fr

[S18]: http://www.gel.ulaval.ca/~vision

[S19]: http://www.ux.his.no/~tranden/brodatz