REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohamed Khider – BISKRA
Faculté des Sciences Exactes, des Sciences de la Nature et de la Vie
Département d'informatique

N° d'ordre : SIOD 16/M2/2018

Mémoire

Présenté pour obtenir le diplôme de master académique en

Informatique

Parcours : **Système Informatique Optimisation Décisionnelle**

Optimisation de la Classification Floue d'image

Par :

KHELIFA MOHAMED AMINE

Soutenu le 26/06/2018, devant le jury composé de :

MEADI Mohamed Nadjib	MAA	Président
CHIGHOUB Fouzia	MAA	Rapporteur
MERABET Rabia	MAA	Examinateur

Remerciement

Remerciement

- Je tiens a remercient tous d'abord seigneur Dieu tout puissant pour m'avoir donné le courage et la santé pour accomplir ce modeste travaille

- Ensuite à présenter ma sincère gratitude à CHIGHOUB FOUZIA pour son dévouement personnel son encadrement, son aide précieuse, et sa patience porté à l'égard de notre projet

- Aussi à remercier le jury pour l'intérêt qu'ils ont bien voulu porter à ce travail en l'honorant par sa présence et son évaluation

- Sans oublié Chouaib Sofiane pour son aide précieuse

- Et pour finir a exprimé toute ma reconnaissance à ma famille, amis et collègues respective pour leur soutien inconditionnel

Dédicace

A ma mère pour son dévouement tous au long de ma vie

A mon père a qui a toujours été à mes cotes

A mon grand frère Khaled pour son soutien et aide inestimable

A ma grande sœur Asma pour sa préoccupation

A tous ceux que mes proche notamment ma grand-mère, Youcef et Randa

A mon unique neveu Samy et mes nièces Mayar et Rinad

A mes cousins : Ramy, Mostefa, Haithem, Zaki

A mes amis : Bachir, Okba, Fayz, Tayeb

A mes collègues : Azza, Baker, Badrane, Rachad

La table des matières

La table des matières

La table des matières

La table des figures

Introduction Générale

Introduction Générale

Aujourd'hui, l'image constitue l'un des moyens les plus importants qu'utilise l'homme pour communiquer avec autrui. C'est un moyen de communication universel dont la richesse du contenu permet aux êtres humains de tout âge et de toute culture de se comprendre. C'est aussi le moyen le plus efficace pour communiquer.

L'image est un ensemble structuré d'informations caractérisé par des paramètres, 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. Le nombre de lignes de cette matrice multipliée par le nombre de colonnes nous donne le nombre total de pixels dans une image.

De ce fait, le traitement d'image est né de l'idée de la nécessité de remplacer l'observateur humain par un ensemble des méthodes et techniques opérant sur celles- ci, dans le but d'améliorer l'aspect visuel de l'image et d'en extraire des informations jugées pertinentes qu'on va utiliser dans différentes applications par exemple la météorologie, l'espace, …et surtout le domaine le plus sensible pour l'être humain à savoir la médecine.

Pour cela le traitement fait appel à des outils, des algorithmes, qui permettent d'agir sur l'image numérisée, L'un des processus fondamentaux dans la chaîne de traitement d'image est la classification.

La classification d'image est une méthode fondamentale, plusieurs méthodes se sont développées ces dernières décennies et les chercheurs ont ressenti le besoin d'avoir une mesure de qualité pour l'évaluation de la classification d'image.

Parmi les algorithmes de classification non-supervisée, Dans les algorithmes de classification floue on trouve l'algorithme C-moyennes flous (FCM), qui nécessite la reconnaissance préalable des nombres des classes, et génère les classes par un processus itératif en minimisant une fonction objective. Qu'il nécessite également une étape d'initialisation des centres, cette dernière rend l'algorithme facilement tombé vers des solutions d'optimum local.

Introduction Générale

Pour essayer de surmonter la limitation mentionnée de la spécification du nombre de classes sous le FCM, deux stratégies sont proposé pour surmontée cela : la première stratégie qui est basé sur les algorithmes génétiques en utilisant la capacité de recherche des algorithmes génétiques, qui s'appuie sur les techniques dérivées de la génétique et des mécanisme d'évolution de la nature : sélection, croissement, mutation, selon la stratégie du survie du plus fort et la deuxième stratégie et celle qui va être adopté durant ce travail est d'énumération qui est lié à la complexité de la structure, pour cela il est exécuté avec différents valeurs initial afin de déterminé le nombre de cluster le plus approprié.

Pour cela le mémoire est composé de 4 chapitres organisé comme suite :

Chapitre 1 : Commence par des généralités sur le traitement d'image et l'explication des notions de base de la segmentation d'images, sa définition, ses domaines d'application, et ses différentes approches.

Chapitre 2 : Contient une présentation détaillée des étapes de l'algorithme FCM elle est composé essentiellement de deux parties, Dans la première partie la définition des algorithmes génétiques avec toutes ses étapes et leur processus d'évolution, tandis que la seconde partie s'oriente vers la stratégie basée sur l'énumération.

Chapitre 3 : Dans la partie de la conception de l'application qui montre l'architecture globale du système ainsi la description détaillée de chaque composant et leurs relations par des schémas et des algorithmes.

Chapitre 4 : Pour finir, la démonstration de l'implémentation de l'application, qui contient l'environnement de développement, les différents composants nécessaires à son fonctionnement, le guide d'utilisation du logiciel à travers les captures d'écran et quelques résultats.

Chapitre I : La Segmentation

Chapitre I : La Segmentation

I.1.Introduction

La segmentation d'images est l'un des problèmes phares du traitement d'images, elle consiste à partitionner l'image en un ensemble de régions connexes. L'intérêt de ces régions est de pouvoir être manipulées ensuite via des traitements de haut niveau pour extraire des caractéristiques de forme, de position, de taille, etc.

Le problème est évidemment très mal posé, car on ne sait jamais dire quelle est la segmentation idéale. L'idée est bien sûr que la région se rapproche de la notion d'objet, au sens courant du terme. Néanmoins, on peut dégager des propriétés plus raisonnables qu'on cherche à obtenir dans un algorithme de segmentation, en particulier : [1]

➢ Stabilité : la segmentation obtenue ne doit pas varier beaucoup lorsque les conditions d'acquisition varient légèrement (bruit, illumination, point de vue...)

➢ Régularité : les régions obtenues doivent être simples à manipuler (taille suffisante, forme régulière...)

I.2.Définition

La segmentation des images constitue le cœur de tout système de vision, c'est une étape importante dans le processus d'analyse des images, c'est un des sujets qui a été le plus étudié dans ce domaine. Elle permet de partitionner une image en un ensemble de points appelés régions, homogènes pour une ou plusieurs caractéristique (intensité, couleur, texture, …) et sont différentes pour au moins une de ses caractéristiques des régions voisines. [2]

Quelques règles à suivre pour obtenir une segmentation sont :

1. Les régions doivent être uniformes et homogènes par rapport à certaines caractéristiques (niveau de gris, écart type, gradient).
2. Leurs intérieurs doivent être simple et sans beaucoup de petits trous (des parties de région non segmentés).
3. Les régions adjacentes doivent avoir des valeurs très différentes par rapport à la caractéristique prise en compte dans la segmentation.
4. Les limites de chaque région doivent être simples et spatialement précises.

La segmentation d'image ainsi définie est un domaine vaste ou l'on retrouve de très nombreuses approches : [3]

- La détection de contours
- La recherche de régions
- L'approche de coopération

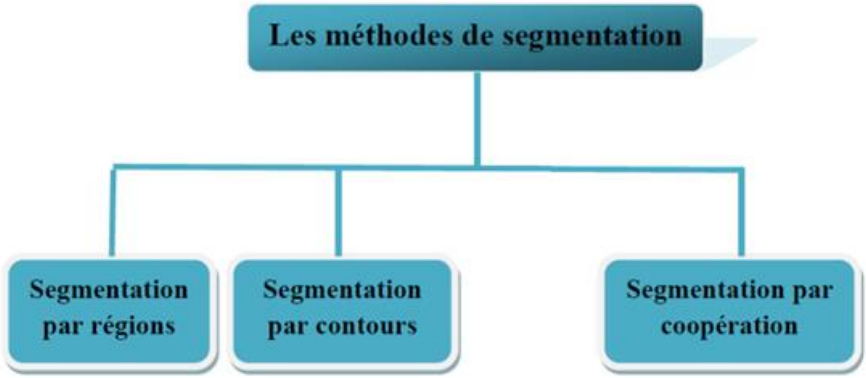

Figure I-1 - Les méthodes de segmentation d'image

I.3.Les Critères de Choix de la Technique de Segmentation

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

I.4.Domaine d'Application de la Segmentation

La segmentation peut être utilisée dans différents domaines. Elle permet de faire une Interprétation automatique des images telle que :

- **Imagerie Médicale**

L'imagerie médicale connait une très forte expansion. Les principales raisons sont:

- L'aide au diagnostic
- La numérisation des données
- La télémédecine

- **Imagerie pour la Sécurité**

De nombreuses applications sont dédiées à la sécurité. On peut citer par exemple :

- Les applications de télésurveillance
- La gestion du trafic en temps réel

- **Imagerie pour la Vision Industrielle**

Pour le contrôle de qualité afin d'éviter le contrôle visuel par un opérateur dont l'objectif est :

- Le contrôle dimensionnel : le système de vision détermine la dimension, la forme et la position de l'objet qu'il observe
- Le contrôle d'aspect : le système détermine la couleur, la texture des objets observés
- Le contrôle de qualité à partir des données précédentes, le système détermine la qualité d'un produit

I.5.Les Différentes Approche de Segmentation

Dans la segmentation nous distinguons 3 approches de Segmentation qui sont :

I.5.1.Segmentation par Approche Contours

La détection de contour est souvent le premier problème qu'on rencontre en traitant une image. La difficulté augmente avec l'importance de bruit présent.

- ***Définition (un contour)***

Un contour peut approximativement être défini comme une frontière entre deux régions ou l'intensité des pixels change brusquement.

Généralement l'utilisation d'un tel operateur de contour se combine avec un seuillage et comme étant ce dernier est généralement imparfait, on obtient, d'une part, des contours qui ne limitent pas les régions fermées. Donc on doit faire recours à des algorithmes de fermetures des contours. D'une autre part, les zones de fortes variations ne correspondent pas forcement à un contour d'objet. Alors un post-traitement est nécessaire pour analyser les différents contours obtenus [3]

- ***La détection de contours***

Les contours sont les lieux de variations significatives de l'information. Supposons que l'image soit une mosaïque de régions parfaitement homogènes, la transition étant stricte, un contour est alors une chaîne de pixels d'épaisseur 1. Cette restriction sur la nature du contour a été imposée dans un premier temps pour des raisons de formalisation mathématique.

Cependant, il n'existe pas à l'heure actuelle de processus complet et général qui pourrait extraire tous les types de contour.

L'extraction de contour est une technique très utilisée dans les domaines scientifiques et techniques, en effet, elle peut aussi bien servir pour la reconnaissance de formes en industrie que pour traiter des images en astronomie afin de les rendre plus claires.

Le principe est donc essentiellement d'effacer tous les motifs à faible variation des niveaux de gris (ou de couleurs) pour ne conserver que les lignes de séparation entre régions homogènes. Il existe pour cela toute une gamme classique d'opérations de filtrage par convolution (notamment le filtre passe haut) (Figure I-2)

- **Toit :** il s'agit d'une ligne sur un fond uniforme.
- **Rampe :** le contour est plus flou.
- **Marche d'escalier :** le contour est net (contour idéal).

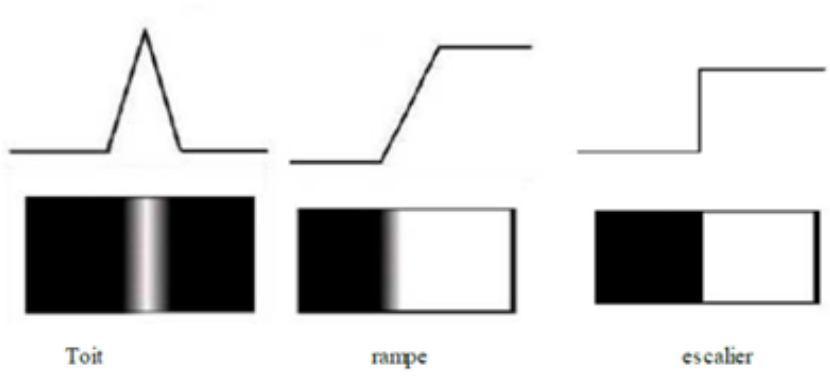

Figure I-2 - Quelque modèles de contours

Nous présentons dans ce qui suit les différentes méthodes adaptées pour la détection des contours dans des images en niveaux de gris. Pour ces dernières, deux familles de méthodes sont distinguées (Figure I-3):

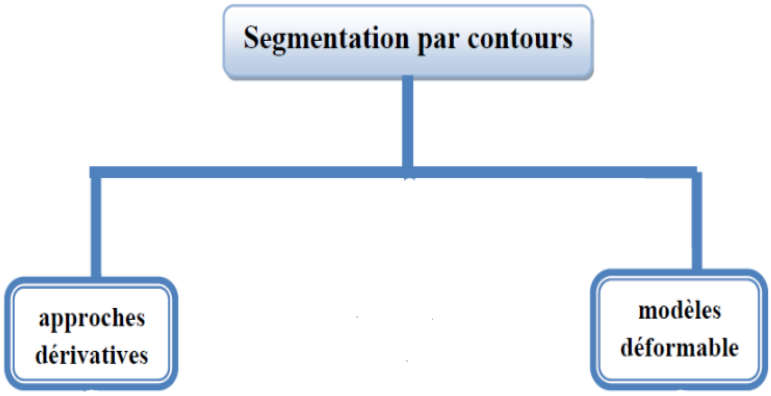

Figure I-3 - Les différentes méthodes de segmentation d'image par contours

I.5.1.1.Les Méthodes dérivatives

Les méthodes dérivatives sont les plus utilisées pour détecter des transitions d'intensité par différenciation numérique. Ce sont des méthodes locales qui balayent l'image avec un masque définissant la zone d'intérêt. À chaque position, un opérateur est appliqué afin de détecter les transitions significatives au niveau de l'attribut de discontinuité choisi. Le résultat est une image binaire constituée de points de contours et de points non-contours. Pour obtenir des régions homogènes il est nécessaire d'utiliser en aval de ces méthodes des traitements qui ferment les contours ouverts et remplissent les zones délimitées par ces contours. Etant donné que les opérateurs de dérivation sont très sensibles au bruit, les images bruitées doivent être lissées au préalable. Le lissage et la dérivation sont en pratique réunie dans un seul filtre.

De nombreuses techniques d'extraction de contours existent dans la littérature. Elles peuvent être classées dans les catégories suivantes :

- Les algorithmes basés sur le gradient (ou opérateurs du premier ordre).
- Les algorithmes basés sur le Laplacien (ou opérateurs du second ordre).

I.5.1.2.Les Méthodes déformables

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

Un contour actif est défini comme une courbe minimisant une énergie et évoluant de manière itérative à partir d'une position initiale proche du contour. Cette approche consiste à combiner les deux étapes citées précédemment à savoir l'extraction et le chainage en une seule étape. Le principe des contours actifs est de faire évoluer un contour initial autour de l'objet d'intérêt vers une position d'équilibre, c'est-à-dire une direction des bords de l'objet à détecter. Entre deux positions différentes du contour (deux itérations), le mouvement des points est fait par une équation impliquant des forces qui agissent sur le contour. Ces forces dépendent des données de l'image (tel que le gradient et l'intensité) et des propriétés du contour (rigidité, élasticité…) [4]

I.5.1.3.Les limites de Segmentation par Contour

Les principales limites des méthodes de détection des contours sont les suivantes [5]

- Les contours extraits selon les méthodes classiques souvent ne correspondent pas nécessairement à la limite des objets. Dans de nombreuses *images de basse qualité,* quelques-unes des méthodes produisent des *faux contours*.
- Les techniques de détection des contours dépendent de l'information contenue dans *le voisinage local* de l'image. Il n'y a pas *d'information globale*.
- Dans la plupart des cas, les stratégies de détection des contours ignorent l'organisation d'ordre supérieur qui peut être utilement présent dans l'image.
- Aprés l'extraction des points de contours, ces derniers sont relies afin de determiner les frontieres. Le processus de fermeture des contours peut parfois conduire à des discontinuites et des lacunes dans l'image.
- Il est souvent difficile d'identifier et de classer les contours parasites.

I.5.2.Segmentation par Approche Région

La segmentation par régions est une approche spécifique dans laquelle on cherche à Construire des surfaces en regroupant des pixels voisins suivants un critère d'homogénéité. Elle peut créer un ensemble de régions qui ont les propriétés suivantes :

- La réunion de toutes les régions donne l'image entière
- Les régions sont connexes (c'est-à-dire que tous les pixels d'une même région sont jointifs)
- Tous les pixels d'une même région sont homogènes entre eux
- Les pixels de deux régions adjacentes ne sont pas homogènes entre eux

Il existe plusieurs méthodes telles que la segmentation par croissance de région, par division de région, et par fusion de région qui sont présenté ci-dessous.

I.5.2.1.Croissance de région (Région Growing)

Cette technique consiste à faire progressivement accroître les régions autour de leur point de départ. L'initialisation de cette méthode consiste à considérer chaque pixel comme une région. On essayer de les regrouper entre elles avec un double critère de similarité :

Niveaux de gris : par exemple la variance des niveaux de gris de la région R est inférieure à un seuil.

Graphe d'adjacence qui est une structure de données qui permet de trouver facilement les régions voisines. (Figure I-4)

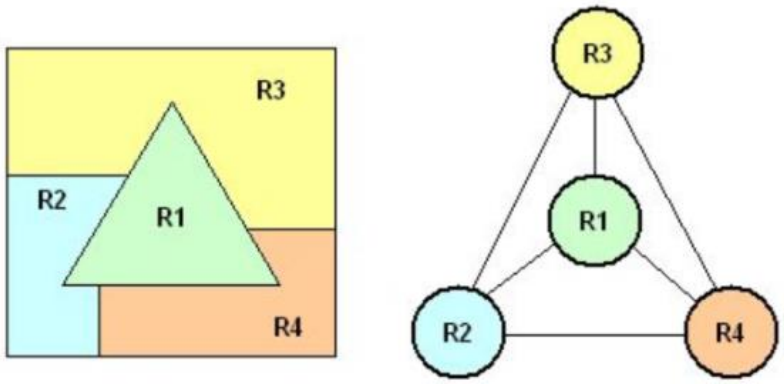

Figure I-4 - Représentation d'un graphe d'adjacence à partir d'un ensemble de régions

Le principe de l'agrégation de pixel est le suivant : on choisit un germe (Le point de départ est le choix d'un ensemble de pixels appelés « germes ») et on fait croitre ce germe tant que des pixels de son voisinage vérifient le test d'homogénéité. Lorsqu'il n'y a plus de pixels candidats dans le voisinage, on choisit un nouveau germe et on itère le processus. (Figure I-5) [6]

Figure I-5 - Croissance progressive de région

Parmi les avantages de cette technique, nous pouvons citer :

- La simplicité et la rapidité de la méthode.
- La segmentation d'objet à topologie complexe.
- La préservation de la forme de chaque région de l'image.

Cependant, il existe plusieurs inconvénients comme :

- L'influence du choix des germes initiaux et du critère d'homogénéité sur le résultat de la segmentation.
- Une mauvaise sélection des germes ou un choix du critère de similarité mal adapté peuvent entraîner des phénomènes de sous-segmentation1ou de sur-segmentation.
- Il peut y avoir des pixels qui ne peuvent pas être classés.

I.5.2.2.Segmentation par fusion de régions (Region Merging)

Les techniques de réunion (*region merging*) sont des méthodes ascendantes où tous les pixels sont visités. Pour chaque voisinage de pixel, un prédicat *P* est testé. S'il est vérifié les pixels correspondants sont regroupés dans une région. . Après le parcours de toute l'image, les groupes de voisinages se voient appliquer le même test, et sont réunis si P est vérifié. Le processus est itéré jusqu'à satisfaction d'un critère d'arrêt

Les inconvénients de cette méthode se situent à deux niveaux :

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

I.5.2.3.Segmentation par division de régions (Split)

La division consiste à partitionner l'image en régions homogènes selon un critère donné. Le principe de cette technique est de considérer l'image elle-même comme région initiale, qui par la suite est divisée en régions. Le processus de division est réitéré sur chaque nouvelle région (issue de la division) jusqu'à l'obtention de classes homogènes (Figure I-6) [7]

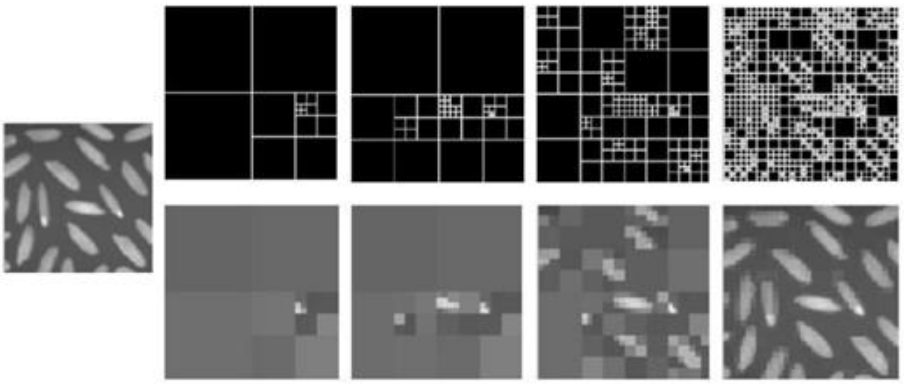

Figure I-6 - Décomposition successives de blocs

Cette méthode présente un inconvénient majeur qui est là sur-segmentation. Toutefois, ce problème peut être résolu en utilisant la méthode de division-fusion que nous présentons dans ce qui suit.

I.5.2.4.Segmentation par division-fusion (Split and Merge)

Ces méthodes combinent les deux méthodes décrites précédemment, la division de l'image en de petites régions homogènes, puis la fusion des régions connexes et similaires au sens d'un prédicat de regroupement. On part du principe que chaque pixel représente à lui seul une région. Deux régions seront fusionnées si elles répondent aux critères de similarité des niveaux de gris et d'adjacence de régions .On s'arrête quand le critère de fusion n'est plus vérifié (Figure I-7) [8]

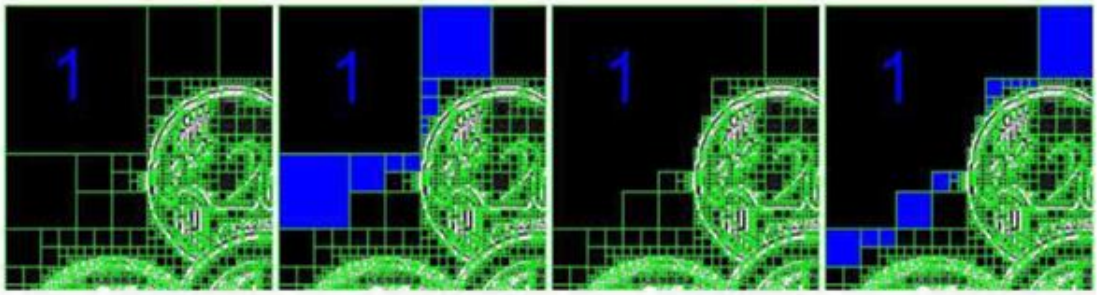

Figure I-7 - Agrégation itérative des blocs similaires au bloc 1

Les inconvénients de cette méthode se situent à trois niveaux :

- Les régions obtenues ne correspondent pas, dans tous les cas, aux objets représentés dans l'image.

- Les limites des régions obtenues sont habituellement imprécises et ne coïncident pas exactement aux limites des objets de l'image.

- La difficulté d'identifier les critères pour agréger les pixels ou pour fusionner et diviser les régions. [8]

I.5.2.5. Segmentation par Classification

Les méthodes de segmentation par classification permettent de regrouper les objets en groupes ou classes plus homogènes, où les objets regroupés ont des caractéristiques communes, ils sont similaires mais se distinguent clairement les objets des autres classes

Autrement dit, ce type de segmentation a pour objectif d'affecter chaque pixel à une classe unique (Figure I-8).

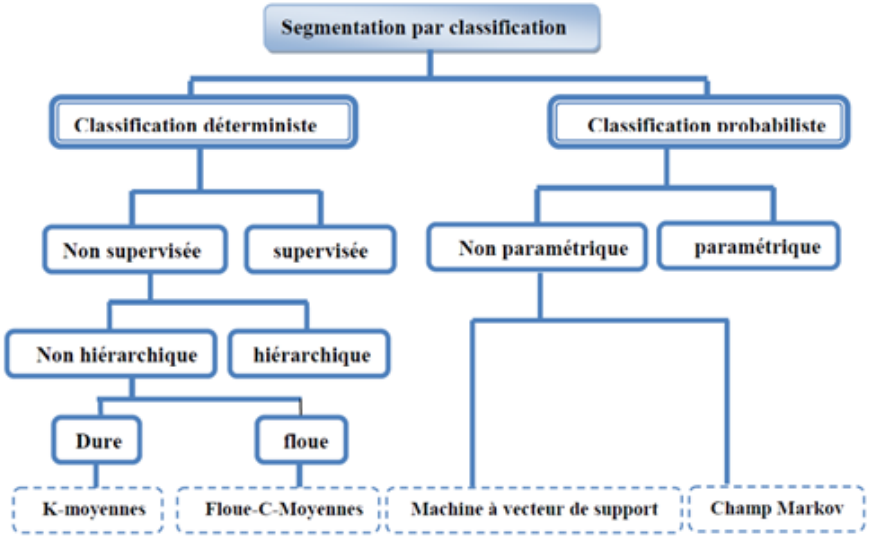

Figure I-8 - Les différentes méthodes de segmentation d'image par classification

- **Classification probabiliste**

  Nous distinguons deux types de classification **probabiliste :**

  - **Classification paramétrique :** Ce type de méthodes s'appuie sur la définition et l'estimation de densités de probabilité des vecteurs de caractéristiques associés aux données, Un label (ou classe, ou hypothèse) est alors attribué à chaque pixel ou voxel d'image, à partir de leur niveau radiométrique.
  - **Classification Non paramétrique :** La particularité des approches probabilistes non paramétriques est leur capacité à discriminer les vecteurs formes en faisant peu d'hypothèses sur les densités de probabilités, Elle contient deux méthodes principaux « Machines à Vecteurs de Supports ou Séparateurs à Vaste Marge (SVM), Champ de Markov CHM ».

- **Classification déterministe**

  Nous distinguons deux types de classification **déterministe :**

  - **Classification supervisée** : Ces sont des méthodes dans lesquelles les classes sont connues a priori avant d'effectuer l'opération d'identification des éléments de l'image. Elles demandent une phase d'apprentissage sur l'échantillon représentatif dans le but d'apprendre les caractéristiques de chaque classe et une autre phase pour décider l'appartenance d'un individu à telle ou telle classe. Parmi les algorithmes de classification supervisés celui de Bayes, Les K plus proches voisins (K-PPV) ou les réseaux de Neurones Multicouches.

- **Classification non supervisée :** Les méthodes non supervisées (Clustering) ne nécessitent pas de données d'apprentissage. Ce sont des méthodes itératives qui, afin de compenser l'absence d'apprentissage, segmentent l'image dans un premier temps puis analysent les caractéristiques de chaque classe afin de réattribuer au mieux les pixels.

Cette dernière se devise en deux familles selon deux approches : (**Approches hiérarchique** et **Approches non hiérarchique**).

- **Classification hiérarchique** : La classification hiérarchique fournit une hiérarchie de partitions. Ce type de classification regroupe les méthodes ascendantes et descendantes. Ces algorithmes essaient de créer une hiérarchie de classes. Les objets les plus similaires sont rassemblés dans des groupes aux plus bas niveaux, tandis que les objets moins similaires se retrouvent dans des groupes aux plus hauts niveaux.

- **Classification non hiérarchique (partitionnelle) :** Il n'y a pas de hiérarchie. Dans la plupart de ces méthodes, le choix a priori du nombre de classes est nécessaire. Les points d'initialisation peuvent être des points du nuage pris au hasard ou les centroides d'une partition préalable. Les classes obtenues n'ont pas plus d'importance les unes que les autres.

Donc, nous présentons la méthode de classification dure K-Means (k moyennes) et nous présentons ensuite, les méthodes de classification floue basée sur la théorie des sous-ensembles flous

- **La méthode de classification K-means**

La classification automatique (ou clustering) est un domaine d'étude situé à l'intersection de deux thématiques de recherches majeures que sont l'analyse de données et l'apprentissage automatique. Ce domaine est en perpétuelle évolution du fait de l'apparition constante de nouveaux besoins portant à la fois sur la quantité ou la nature des données à traiter (numériques, symboliques, spatiales, histogrammes, etc.) que sur le type de classification attendue (partition, hiérarchie, schéma flou, etc.).

L'algorithme K-means, également appelé algorithme des nuées dynamiques, est l'algorithme de clustering le plus connu et le plus utilisé du fait de sa simplicité de mise en œuvre. Il partitionne les données d'une image en K clusters. L'algorithme renvoie une partition des données, dans laquelle les objets à l'intérieur de chaque cluster sont aussi proches que possible les uns des autres et aussi loin que possible des objets des autres clusters. Le résultat est un ensemble de clusters compacts et séparés, si la valeur K du nombre du cluster est bien choisie au départ.

Les principales étapes de l'algorithme K-means sont (Figure I-9):

1. Choix aléatoire de la position initiale des K clusters.
2. (Ré-) Affecter les objets à un cluster suivant un critère de minimisation des distances (généralement selon une mesure de distance euclidienne).
3. Une fois tous les objets placés, recalculer les K centroïdes.
4. Réitérer les étapes 2 et 3 jusqu'à ce que plus qu'aucune réaffectation ne soit faite.

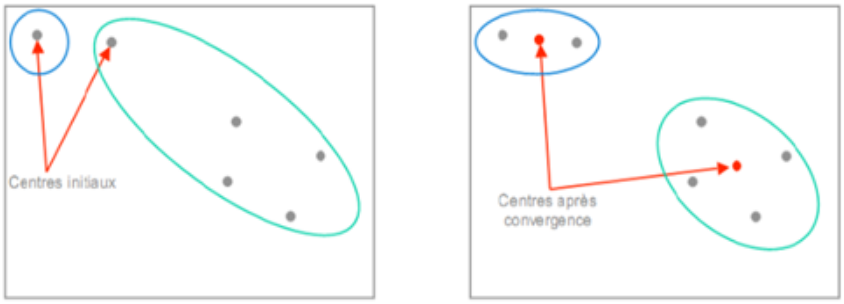

Figure I-9 - Principe de fonctionnement général de k-means

Cette méthode présente des inconvénients majeurs du fait que le nombre de clusters doit être choisi à l'initialisation. La classification finale est étroitement liée au nombre de classes initial (Figure I-10).

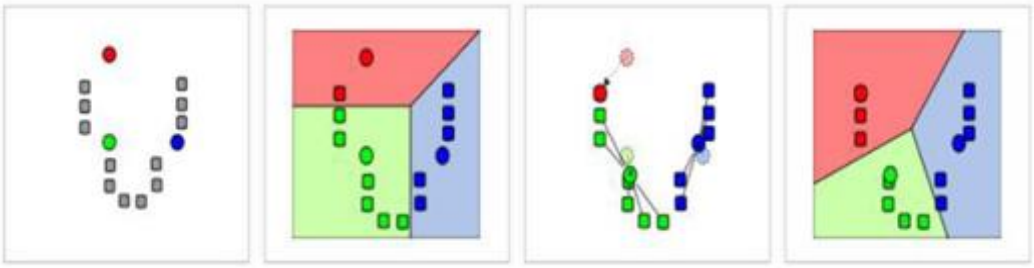

Figure I-10 - Démonstration des quatre étapes de k-means avec (K=3)

La méthode des *k*-means a été très utilisée, d'une part pour sa simplicité et sa rapidité, en effet, le nombre d'itérations est typiquement inférieur au nombre de points. Et d'autre part car elle peut fournir une bonne approximation de la segmentation recherchée est extrêmement rapide en pratique. Néanmoins, l'utilisateur doit choisir le nombre *k* de classes et il n'y a pas de critère unique pour déterminer le meilleur nombre de classes, Lorsque les classes ne sont pas bien séparées, les *k*-means tendent à trouver uniquement des classes "sphériques", de même effectif et de même volume. Plusieurs initialisations peuvent conduire à plusieurs partitions très différentes dans leur composition, mais très proches et quasi-optimales au sens de la fonction objective, c'est-à-dire cet algorithme ne garantit pas un optimum global. Le *k*-means souffre aussi d'un défaut qui a son importance en segmentation d'images : elle introduit des discontinuités spatiales assez fortes aux frontières des classes.

- **La méthode de classification Fuzzy C-Means**

L'algorithme FCM est une méthode de classification non supervisée très populaire et très performante par rapport aux autres méthodes comme l'algorithme K-means ou les réseaux de neurones

Il est abondamment utilisé pour la segmentation d'images

Malgré cela, l'algorithme FCM souffre de certains inconvénients comme sa sensibilité au bruit, la forme des classes qui sphérique, la nécessité de connaitre le nombre de classes ainsi que la dépendance de ses résultats à l'initialisation. Pour surmonter ces inconvénients plusieurs variantes du FCM ont été proposées

Le fuzzy C-means est un algorithme de classification non supervisée, issu de l'algorithme K-means. Son apport par rapport à ce dernier est l'introduction de la notion du flou, afin de prendre en compte l'imprécision des données. L'algorithme FCM est un algorithme de réallocation floue, dans lequel les classes sont représentées par des prototypes (centres de gravité). Son application fournit donc pour chaque observation à classifier un degré d'appartenance (compris entre 0 et 1) à chaque classe, produisant ainsi une partition floue. Comme pour la plus part des autres algorithmes de classification par partition, le FCM est basé sur la minimisation d'un critère en suivant un processus itératif

On peut maintenant résumer le processus de l'algorithme FCM sous les étapes suivantes :

1. L'initialisation de nombre de classes
2. L'initialisation des centres
3. Le calcul de la matrice d'appartenance
4. Recalcule les centres
5. Calcul du critère d'arrêt et retour à l'étape 2 s'il y a non convergence de critère

I.5.2.6.Les Limites de la segmentation par région

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

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

I.5.3.Segmentation par Approche Coopérative

La segmentation par coopération région-contours suscite un grand intérêt ces dernières années. Elle consiste en une coopération entre la segmentation par régions et la segmentation par contours. Elle exploite les avantages de ces deux types de segmentation pour aboutir à un résultat de segmentation plus précis et plus fidèle que celui obtenu à l'aide d'une seule technique [9]

Jusqu'à présent, on a vu qu'il existait plusieurs approches de segmentation d'images, chacune agit de manière différente et utilise des attributs différents. Dans certains cas, le choix de la méthode n'est pas évident et les paramètres à utiliser peuvent être nombreux. De plus, chaque méthode possède ses avantages et ses limites d'utilisation suivant le problème à résoudre.

Le but de coopérer les méthodes de segmentation est d'utiliser les avantages de plusieurs méthodes en même temps. La coopération peut être associative ou séquentielle (par contraintes) [10]

I.5.3.1.Coopération séquentielle

Le principe général de la coopération séquentielle est que l'une des techniques, par régions ou par contours, est réalisée en premier lieu. Son résultat sera par la suite exploité par l'autre technique. L'intégration de l'information provenant de la segmentation par contours dans une segmentation par régions est l'une des formes de coopération les plus courantes. Mais, l'information sur les régions peut aussi être intégrée dans une segmentation par contours (Figure I-11) [11]

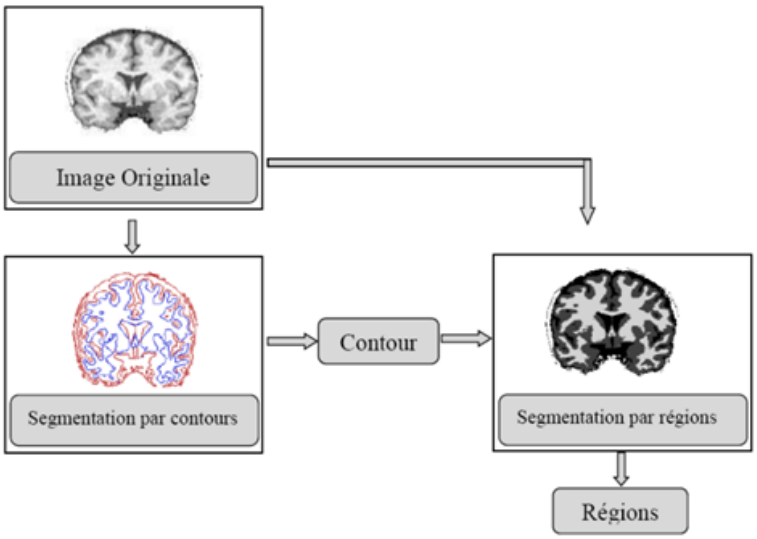

Figure I-11 - Principe de la Coopération séquentielle Régions-Contours

I.5.3.2. Coopération des associatives (parallèle)

Dans la coopération des associative ou résultats, les deux types de segmentations sont réalisés d'une façon parallèle et indépendante, et la coopération sera faite au niveau de leurs résultats respectifs. Ils seront intégrés dans le but d'atteindre une meilleure segmentation que celle obtenue par une seule des techniques (Figure I-12).

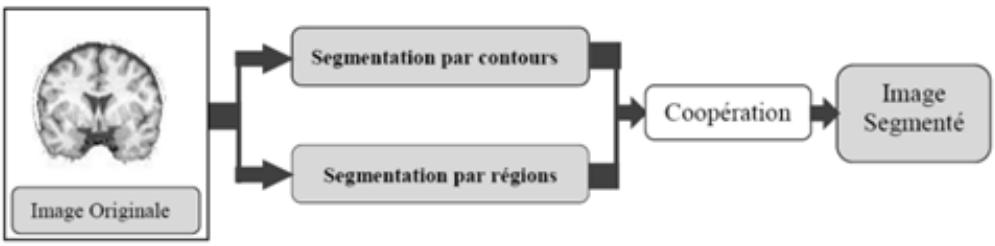

Figure I-12 - Principe de la coopération associative Régions-Contours

I.6. Conclusion

A travers ce chapitre on a vu qu'il existe plusieurs méthodes pour la segmentation par région et par contour. Bien elles aient toutes des avantages, elles présentent également plusieurs inconvénients. Pour cela, plusieurs chercheurs se sont intéressés aux approches coopératives où elle tire davantage de la dualité des deux approches pour une meilleure segmentation. Dans le chapitre suivant, on va présenter un état de l'art sur les méthodes de segmentation par classification et en particulier la classification floue qui nécessite la reconnaissance préalable des nombres des classes.

Pour cela on propose deux méthodes qui de déterminé le nombre de classes : celle basé sur les algorithmes génétique et celle d'énumération.

Chapitre II : La Spécification du nombre de Classes sous le FCM

Chapitre II : La Spécification du nombre de Classes sous le FCM

II.1.Introduction

La classification est une activité mentale qui intervient fréquemment dans la vie courante. En effet, les objets sont souvent répertoriés par rapport à des classes ou des catégories auxquelles ils sont censés appartenir. Cette appartenance est, la plupart du temps, vague et/ou graduelle. Des modificateurs linguistiques tels que très, trop, assez, insuffisamment, traduisent l'incertitude que l'on peut rencontrer concernant les caractéristiques ou les propriétés de certaines entités.

Dans le cadre de ce chapitre on va expliquer la classification et ses méthodes (supervisé et non supervisé), parmi les méthodes non supervisé il y a l'algorithme FCM, ou on s'intéresse aux travaux relative au problème de spécification de nombre de classes de celui-ci par deux stratégie : génétique et énumération

II.2. La classification

La classification consiste à attribuer à chaque pixel dans l'image une classe (étiquette). Cette affectation peut être effectuée on se basant sur des régions dont on connait les classes d'appartenance a priori, alors, on parle de classification supervisée, ou non, dans ce cas on parle de classification non-supervisée (automatique) [12].

II.3. Les méthodes de classification

Les problèmes de classification s'attachent à déterminer des procédures permettant d'associer une classe à un objet (individu). On peut classer les méthodes de classification en deux grandes classes : la classification supervisée et la classification non supervisée.

II.3.1. La classification supervisée

Les méthodes de classification supervisées supposent que les classes sont connues a priori avant d'effectuer l'opération d'identification des éléments de l'image. Elles demandent une phase d'apprentissage sur l'échantillon représentatif dans le but d'apprendre les caractéristiques de chaque classe et une autre phase pour décider l'appartenance d'un individu à telle ou telle classe [13].

II.3.2. La classification non supervisée (automatique)

La méthode de classification non supervisée a pour but de trouver des partitions d'un ensemble d'individus en fonction de critères de proximité de leurs vecteurs d'attributs dans l'espace de représentation. Elles sont utilisées pour effectuer une classification en aveugle et, ainsi pour réaliser une segmentation sans connaissances a priori sur l'image.

Parmi les algorithmes de classification non supervisés on trouve celui des centres mobiles (K-means), et le C-moyennes floues ("Fuzzy C-Means" ou FCM) qu'on va l'expliqué plus en détails

II.4.Fuzzy C-Means

Fuzzy C-Means (FCM) est une méthode de classification non supervisé populaire qui a été largement appliquée pour la segmentation d'images qui s'appuie sur la logique floue, ou une donnée peut appartenir à deux ou plusieurs classes quand le nombre de classes est connu a priori.

II.4.1.Principe

Fuzzy C-Means (FCM) généralise l'algorithme K-Means, introduit la notion d'ensemble flou dans la définition des classes : le fait qu'un pixel possède aussi bien des attributs qui l'assignent à une classe qu'à une autre. Il assigne donc, non pas à un pixel une étiquette relative à une classe unique, mais son degré d'appartenance à chacune des classes. Ces valeurs expriment l'appartenance incertaine d'un pixel à une région et sont appelées degrés d'appartenance. Le degré d'appartenance se situe dans l'intervalle [0,1] et les classes obtenues ne sont pas forcément disjointes. Dans ce cas, les données $\boldsymbol{x_j}$ ne sont plus assigné par l'intermédiaire unique, mais à plusieurs par l'intermédiaire de degrés d'appartenance $\boldsymbol{u_{ij}}$ du vecteur $\boldsymbol{x_j}$ à la classe $\boldsymbol{i}$. Le but est non seulement de calculer les centres de classe $\boldsymbol{C}$ mais aussi l'ensemble des degrés d'appartenance des vecteurs aux classes.

Si $\boldsymbol{U_{ij}}$ est le degré d'appartenance de $\boldsymbol{x_j}$ à la classe $\boldsymbol{i}$, la matrice $\boldsymbol{U_{CxN}[u_{ij}]}$ est appelée matrice de C-partitions floues si et seulement si elle satisfait aux conditions (II-1)(II-2) :

$$\forall i \in \{1..C\}, \forall j \in \{1..N\} \begin{cases} U_{ij} \in [0,1] \\ 0 < \sum_{j=1}^{N} U_{ij} < N \end{cases} \qquad \text{(II-1)}$$

$$\forall j \in \{1..N\} \sum_{i=1}^{C} U_{ij} = 1 \qquad \text{(II-2)}$$

Le partitionnement flou du FCM est réalisé par une minimisation de la fonction objective, La minimisation de la fonction objective ***J(C, U, X)*** nécessite une mise à jours itérative des centres et des degrés d'appartenance jusqu'au le critère d'arrêt est établi [14]. La fonction objective à minimiser ***J*** est **:**

$$J(C, U, X) = \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij}^{m} \left\| x_j - c_i \right\|^2 \quad \text{(II-3)}$$

La minimisation de la fonction objective nécessite une mise à jour itérative des centres et de la matrice des degrés d'appartenance par les deux formules suivantes :

$$c_i = \frac{\sum_{k=1}^{n} U_{ik}^{m} . X_k}{\sum_{k=1}^{n} U_{ik}^{m}} \quad \text{(II-4)}$$

$$U_{ij} = \left[\sum_{k=1}^{C} \left(\frac{d^2(x_j, b_i)}{d^2(x_j, b_k)} \right)^{\frac{2}{(m-1)}} \right]^{-1} \quad \text{(II-5)}$$

- ***m*** : le coefficient de fuzzification ϵ [1.5 - 2.5].
- $\left\| \boldsymbol{x_j} - \boldsymbol{c_i} \right\|^2$La distance Euclidienne entre le pixel j et la classe i.
- $\boldsymbol{U_{ij}}$**:** le degré d'appartenance de pixel j à la classe i.
- $\boldsymbol{C_i}$**:** le centre du classe i.
- ***C*** : le nombre des classes
- ***N*** **:** le nombre total des pixels de l'image.

II.4.2.Le processus de l'algorithme FCM

L'algorithme FCM peut être résumé comme suit :

1. Fixer les paramètres :
    a. Le nombre de classes *K*
    b. Le seuil $\mathcal{E}$ représentant l'erreur de convergence
    c. Le degré flou *m*, généralement pris égal à 2
    d. t étant la $ti\grave{e}m$ itération
2. Initialiser les centres des K classes de manière aléatoire
3. Mettre à jour le vecteur V des centres des classes par

$$c_i = \frac{\sum_{k=1}^{n} U_{ik}^{m} . X_k}{\sum_{k=1}^{n} U_{ik}^{m}} \quad \text{(II-4)}$$

4. Mettre à jour la matrice *U* des degrés d'appartenance par la

$$U_{ij} = \left[\sum_{k=1}^{C}\left(\frac{d^2(x_j,b_i)}{d^2(x_j,b_k)}\right)^{\frac{2}{(m-1)}}\right]^{-1} \qquad \text{(II-5)}$$

5. Répéter les étapes 3 et 4 jusqu'à satisfaction du critère d'arrêt

$$\left|\left|V^{(t)} - V^{(t+1)}\right|\right| < \varepsilon \qquad \text{(II-6)}$$

II.4.3. Critère d'arrêt et terminaison

L'algorithme FCM s'arrête lorsque le critère d'arrêt est vérifié, soit après certain nombre d'itérations défini au début de l'algorithme ou lorsque les centres des classes n'évoluent plus entre deux itérations successives, c'est-à-dire la différence entre les centres de deux itérations successives ne dépasse pas certain seuil prédéfini.

A la fin d'appliquée l'algorithme FCM, ce dernier nécessite le passage par une étape d'étiquetage. L'objectif de cette étape est la construction de l'image finale à partir de la matrice des degrés d'appartenance obtenu après la convergence de l'algorithme de classification, et pour réaliser cette étape on affecte chaque point à la classe du degré d'appartenance le plus élevé

II.5. La spécification du nombre de classes sous le FCM

L'algorithme FCM a été largement étudié et a été utilisé dans bon nombre de domaines (segmentation d'images médicales, géologiques et satellitaires). Cependant cet algorithme qui nécessite la connaissance préalable du nombre de classes, n'est pas robuste face aux bruits introduits par l'imprécision des attributs et son efficacité dépend fortement de l'étape d'initialisation des centres des classes car le processus itératif peut facilement fournir une solution localement optimale. De plus, il est basé sur la distance euclidienne lors de la mesure de similarité entre une observation et le centre d'une classe ce qui le rend utilisable que pour détecter des classes de forme sphérique. Afin d'éviter ces inconvénients donc améliorer les résultats de la classification, plusieurs modifications ont été apportées à l'algorithme standard et qui sont présentées comme des variantes du FCM [15]

II.5.1. Stratégie basée sur l'Algorithme Génétique

Plusieurs travaux ont adopté une stratégie basée sur les algorithmes génétiques. Les auteurs dans le travail [16] utilisent Les Algorithmes Génétiques de chaîne de longueur variable qui s'appuie sur les techniques dérivées de la génétique et des mécanismes de l'évolution de la nature, où des chromosomes différents d'une même population peuvent coder un nombre différent de cluster. Les auteurs utilisent l'indice d'évaluation des partitions floues Xie-Beni [16] pour mesurer les valeurs de fitness des chromosomes.

L'ensemble U de toutes les possibles matrices de partition floue est représenté par

$$U = \{U \in \mathbb{R}^{c*n} | \sum_{i=1}^{c} u_{ik} = 1, 0 < \sum_{k=1}^{n} u_{ik} < n, \ \ et\ u_{ik} \in [1,0]\} \qquad \text{(II-7)}$$

La meilleure partition est celle qui correspond à la valeur minimale de l'indice Xie-Beni *(XB (U, V, X))* [16], où *U, V et X* sont respectivement la matrice de partition, l'ensemble des centres de cluster et l'ensemble des données d'entrée. En d'autres termes, la meilleure matrice de partition U^* peut être représentée comme :

$$U^* \in \text{u}\ et\ XB(U,V,X) = \min XB(Ui,Vi,X) \qquad \text{(II-8)}$$

Où V_i représente l'ensemble des centres de clusters correspondant à U_i.

Les chromosomes sont constitués des nombres réels qui représentent les coordonnées des centres des partitions. Si le chromosome i code les centres des clusters Mi en espace N de dimension alors sa longueur li est prise pour être $N*M_i$. Chaque chaîne i de la population initiale code les centres d'un nombre M_i de clusters, tel que $M_i = (rand()\ mod\ M^*)+2$. Ici, *rand()* est une fonction qui renvoie un entier, et *M** est une estimation de la limite supérieure du nombre de clusters.

Le nombre de clusters sera donc compris entre deux et *M*+1*. Notez que *M** n'est utilisé que pour la génération de la population initiale. Le nombre réel de clusters dans l'ensemble de données n'est pas lié à *M**, et peut être n'importe quel nombre supérieur, égal ou inférieur à *M**. La M_i des centres codés dans un chromosome sont choisis de manière aléatoire parmi des points distincts de l'ensemble de données.

Le fitness d'un chromosome indique le degré de goodness de la solution qu'il représente. L'indice *XB* [16] est défini comme une fonction du rapport de la variation totale σ à la séparation minimale sep des clusters. Ici σ et sep peuvent être écrits comme

$$\sigma(\mathrm{U},\mathrm{V};\mathrm{X}) = \sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^{2}\, D^{2}(v_i, x_k) \qquad \text{(II-9)}$$

$$sep(V) = \ min_{i\neq j}\{||v_i - v_j||^{2}\} \qquad \text{(II-10)}$$

Où ||.|| est la norme euclidienne, et $D(v_i, x_k)$ est la distance entre le motif x_k et le centre de la classe v_i. L'index *XB* est alors écrit comme

$$XB(U,V;X) = \frac{\sigma(\mathrm{U},\mathrm{V};\mathrm{X})}{n\ sep(V)} = \frac{\sum_{i=1}^{c}(\sum_{k=1}^{n} u_{ik}^{2} D^{2}(v_i, x_k))}{n(min_{i\neq j}\{||v_i, x_k||^{2}\})} \qquad \text{(II-11)}$$

Lorsque le partitionnement est compact, la valeur de σ devrait être faible pendant que sep devrait être élevé, donnant ainsi des valeurs plus faibles de l'indice *Xie-Beni* L'objectif est par conséquent est de minimiser l'indice *XB* afin d'obtenir un regroupement approprié.

La sélection est appliquée sur la chaine de population. Ainsi que l'opérateur de croisement sur les points de croisement de manière stochastique, ou chaque position de gêne d'un chromosome est soumise à une mutation.

L'algorithme est exécuté pour un nombre fixé de générations. La meilleure chaîne de la dernière génération fournit la solution au problème du clustering.

II.5.2.Stratégie basée sur l'Enumération

Le nombre du cluster est un paramètre lié à la complexité de la structure du cluster. Le principe de la stratégie d'énumération est d'exécuté l'algorithme de clustering FCM avec différentes valeurs initiales des centres, et chaque partition floue obtenue s'attribue par une valeur d'évaluation, et le nombre des centres de la partition de la meilleur évaluation correspond au nombre de cluster le plus approprié.

L'algorithme décrit par les auteurs [17] est appelé un algorithme de division, car il fonctionne par diviser le "pire" cluster à chaque étape en testant le nombre des clusters *c* de C_{min} à C_{max}. La stratégie générale adoptée est à chaque étape nous identifions "Pire" cluster et le diviser en deux clusters tout en gardant les autres clusters *c-1*. Cette dernière étape implique l'augmentation de nombre de clusters par un, ce processus se répète jusqu'à l'arrivée au nombre maximal des clusters (C_{max}).

Ici on essaye de déterminé la bonne valeur de C par le critère de validité du cluster pour cela on applique le FCM pour $C=C_{min}$ jusqu'à C_{max} qui sont prédéfinie entre le nombre de classe recherché

(1) Choisir C_{min} *et* C_{max}

(2) Pour $c = C_{min}$ *à* C_{max}

(2.1) Initialisation des centres du cluster *(V)*

(2.2) Appliquer l'algorithme FCM basique pour mettre à jour le degré d'appartenance de la matrice *(U)* et le centre du cluster *(V)*

(2.3) Testez pour la convergence, sinon, allez à (2.2)

(2.4) Calculez la valeur de validité $V_d(c)$

(3) Calculez cf tel que l'index de validité du cluster $V_d(cf)$ est optimal

L'idée générale des auteurs dans [17] est la définition du critère d'identification du "pire" cluster. Dans ce cas une fonction "score" *S(i)* associée à chaque cluster i est définie comme suite :

$$S(i) = \frac{\sum_{k=1}^{n} u_{ki}}{nombre\ des\ vecteurs\ de\ données\ dans\ le\ cluster\ i} \qquad \text{(II-12)}$$

En général, lorsque *S(i)* est petit, le cluster *i* tend à contenir un grand nombre de vecteurs de données avec des valeurs d'appartenance faibles.

Plus la valeur d'appartenance est basse, plus l'objet est éloigné à partir de son centre de cluster. Par conséquent, un petit *S(i)* signifie que le cluster *i* est grand en volume et peu dispersé. Ce qui est la raison pour laquelle nous choisissons le cluster correspondant au minimum de *S(i)* en tant que candidat à partager lorsque la valeur de c est augmentée. D'un autre côté, un plus grand *S(i)* tend à signifie que le cluster i a un plus petit nombre d'éléments et exerce une forte "attraction" sur eux.

(1) Choisir C_{min} et C_{max}

(2) Initialisé $Cmin$ centre cluster (V)

(3) Pour $c = C_{min}$ à C_{max}

(3.1) Appliquer l'algorithme FCM Basique pour mettre à jour le degré d'appartenance (U) et le centre du cluster *V*

(3.2) Tester pour la convergence, sinon, aller a (3.1)

(3.3) Calculer la valeur de validité $V_d(c)$,

(3.4) Calculer le score *S(i)* pour chaque cluster, diviser le pire cluster

(4) Calculez cf tel que l'indice de validité du cluster $V_d(cf)$ est optimal

Afin de diviser le cluster à l'étape 3.4, les auteurs dans [17] ont adapté la technique "Greedy" qui vise à initialiser les centres du cluster aussi loin que possible de l'un à l'autre. De manière itérative, le "Greedy" est une technique qui sélectionne comme un nouveau centre de cluster le vecteur de données qui a la plus grande distance totale des centres de cluster existant. L'adaptation de la technique pour la division en clusters donne l'algorithme suivant :

(1) Identifier le cluster a divisé. Supposons que le numéro de cluster est i_0, son centre et l'ensemble de toutes les données du cluster sont notées V_{i0} et E.
(2) Rechercher E pour le vecteur de données non étiqueté "testé" a la distance totale maximale de tous les autres c-1 centres de cluster. Ce vecteur de données est noté V_{i1}.
(3) Partition E en $E0$ et $E1$ en fonction de la distance de chaque vecteur de données de V_{i0} et V_{i1}. Si $|E1|/|E2|>10\%$, alors V_{i1} est pris comme le centre du cluster du c^{ime}, sinon étiqueter V_{i1} "Testé" et passez à l'étape 2.
(4) Rechercher E pour le vecteur de données non étiqueté "testé" a la distance totale maximale de tout le cluster c centres. Ce vecteur de données est noté V_{i2}.
(5) Partition E en $E1$ et $E2$ en fonction de la distance du vecteur de données de V_{i1} et V_{i2}. Si $|E1|/|E2|>10\%$, alors V_{i2} est pris comme le centre de cluster $C+1$, sinon label V_{i2} "testé" et passez à l'étape 4.

Ce processus se répète jusqu'au $C=C_{max}$, Ou les différentes partitions floues obtenues seront évaluées en termes de qualité par l'indice de validité et la partition optimale qui minimise cet indice sera adoptée.

Une autre approche est proposée par [18], l'idée est de poursuivre un processus qui essaie d'adapter le nombre de cluster K. Ainsi, à partir de $K = K_0$, et à chaque itération la méthode consiste en un test qui vérifie si le modèle de cluster peut être amélioré en augmentant ou en diminuant jusqu'à ce que K reste inchangé pour deux itérations successives, et le FCM sera appliqué pour les trois nombres de classes *(K-1, K et K+1).* Une fonction de qualité $Q(.)$ pour mesurer la qualité des partitions floues est obtenue par :

$$\mathrm{K} \leftarrow \operatorname{argmax}\{\mathrm{Q}(\mathrm{K}-1), \mathrm{Q}(\mathrm{K}), \mathrm{Q}(\mathrm{K}+1)\} \quad \text{(II-13)}$$

Pour passer de K *à* K-1 clusters l'un des clusters est supprimé, et M itérations de l'algorithme des C-moyens floues sont exécutées afin de réattribuer les éléments de ce cluster aux centres de cluster restants, Cette opération est répétée K *fois*. Le meilleur modèle de cluster est alors choisi selon la valeur de la fonction d'évaluation $Q\ (K\text{-}1)$.

Le passage de K à K+1 clusters impose qu'un cluster supplémentaire sera émergé. Pour créer ce cluster un centre parmi les K centres existants est défini par un élément choisi au hasard. La probabilité qu'un élément soit sélectionné est définie comme une fonction croissante de la distance des éléments à partir du centre de cluster associé. En essayent un nombre fixe d'éléments choisis au hasard et de sélectionner celui qui donne le meilleur modèle de cluster.

Les auteurs dans [19] proposent une autre approche de combiner la classification hiérarchique avec le FCM, chaque élément de l'ensemble de données sera considéré comme un cluster. Dans chaque processus de fusion, la somme de l'erreur carrée de tous les éléments de données à leur centre de cluster est calculée, où la distance entre objets est mesurée par la distance euclidienne. Cette dernière reflète la compacité du cluster, donc dans chaque processus de fusion cette méthode fusionne deux clusters qui peuvent conduire au minimum variance de cluster après la fusion.

Le processus de division du clustering est défini comme suit :

Un ensemble $C_i(i=1,2,...,k)$, ou C_i est un cluster, $W_i(i=1,2,...,k)$ ou W_i est la variance cluster du i^{eme} cluster. Diviser le cluster C_i en C_i^1, C_i^2, la variance du cluster d'entre eux sont W_i^1, W_i^2 Et la diminution de la variance de cluster est définie comme suivre :

$$\delta(\mathrm{C_i}) = \mathrm{W_i} - \mathrm{W_i^1} - \mathrm{W_i^2} = \sum ||\mathrm{x_i} - \bar{\mathrm{x}}_\mathrm{i}||^2 - \sum ||\mathrm{x_i^1} - \bar{\mathrm{x}}_\mathrm{i}^1||^2 - \sum ||\mathrm{x_i^2} - \bar{\mathrm{x}}_\mathrm{i}^2||^2 \qquad \text{(II-14)}$$

Ou x_i, x_i, x_i indique le centre de C_i, C_i^1, C_i^2

Le cluster $\mathrm{C_i}$ est choisi pour diviser, par l'application de l'algorithme FCM ce processus est répété jusqu'à obtenir le maximum Ci. Pour deux clusters C_i, C_j. *Si* $\delta(C_i)>\delta(C_j)$, alors une plus petite variance de cluster résulte de la scission de C_i.

Un nouvel indice de variance de cluster est proposé :

$$S(C) = \frac{P(C-1)-P(C)}{P(C)-P(C+1)} \qquad \text{(II-15)}$$

$$P(C) = \sum_{i=1}^{C} W_i \qquad \text{(II-16)}$$

L'Equation (II-15) indique le ratio entre les variations de la variance des clusters dans deux processus de division adjacents. Quand le ratio atteint le maximum, il indique un tournant de la variation de la variance du cluster, et C l'optimum du nombre des clusters.

Le nombre optimal des clusters est celui qui conduit au maximum d'équation (II-15)

Le but du travail de [20] est de produire de bonne partition en utilisant une mesure de validité, ce processus commence avec une partition générer itérativement, en retenant que les modifications qui produisent des partitions avec une meilleure validité, Afin de garder le nombre de cluster constant l'opération de division est accompagnée d'une fusion des deux clusters les plus proches, selon l'opération de fusion qui donne lieu à une partition de meilleure validité. Si l'opération donne une partition avec une meilleure validité, cette partition est conservée, Sinon, la partition précédente est restaurée. Lorsque tous les clusters ont été considérés pour la décomposition, ce processus est répété jusqu'à ce que plus aucune augmentation de l'indice de validité ne soit atteinte

II.6.Conclusion

A travers ce chapitre, on a fourni les notions de base de La classification et ses méthodes (supervisée et non supervisée), et quelques exemples de chaque catégorie

Parmi les méthodes non supervisé on s'est intéresse à l'algorithme FCM (Fuzzy C-Means) notamment son principe et son fonctionnement, avec lequel on a présenté également quelques travaux de spécification de nombre de classes qu'on peut les classer en deux catégories : des méthodes qu'utilisent les stratégies basées sur les Algorithmes Génetiques, et d'autres utilisent les stratégies d'énumération. Et dans le chapitre suivant nous allons présenter durant la conception de notre système.

Chapitre III : La Conception du Système

Chapitre III : La conception du système

III.1.Introduction

La conception est un processus de définition du futur d'une application informatique, Cette phase permet de séparer les problèmes complexes en sous problèmes pour trouver les solutions facilement. Ainsi, elle permet la décomposition modulaire du logiciel pour faciliter la maintenance et la réutilisation, et le développement parallèle.

Les parties mentionnées dans le premier et deuxième chapitre ont permis de comprendre et de situer clairement les notions de base pour la conception et la réalisation du projet. Ainsi dans le processus de développement du système, la présentation de la conception va décrire d'une manière non ambiguë notre système. Nous allons donner l'architecture globale de notre système et ceci selon une vue interne (structures et comportements des composants), puis les détails des fonctionnalités de cette architecture avant de présenter sa réalisation.

III.2.La Conception Globale

Cette partie définit l'architecture globale du système qui se présente à une image comme entrée et avec comme résultat finale une image classée après avoir appliqué la méthode de classification floue par la stratégie d'énumération en utilisant l'algorithme de fusion des régions, ou le résumer de l'architecture globale du système est dans le schéma ci-dessus (Figure III-1)

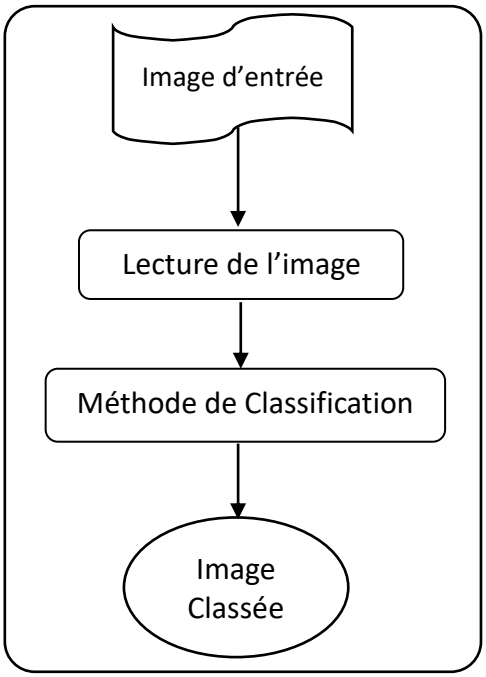

Figure III-1 – Architecture Global du Système

L'Entrée de notre système est une interface qui permet à l'utilisateur de choisir une image comme une entrée sous format JPEG, JPG, PNG, TIF.

III.2.1.La méthode de classification

La méthode utilisée pour la classification d'image dans ce travail est une méthode de classification floue (FCM) combiné avec la stratégie d'énumération, afin de spécifier le nombre de classes pour arriver à une image classifiée.

Le principe de cette dernière est :

Premièrement une étape d'initialisation des centres, pour appliquer l'algorithme FCM sur les centres obtenue $C=C_{max}$, Ensuite vient l'étape de fusion, qui pour chaque cluster on calcul son fitness, cette dernière est définie par la variation à l'intérieur du même cluster et la séparation aux autres clusters, le cluster de fitness maximale sera le cluster à éliminer, qui a pour résultat la décrémentation du nombre de clusters par un (C=C-1), on répète le calcul des nouveaux centres qui seront les entrées du FCM, après la convergence une évaluation de la partition floue obtenue est faite, ce processus de fusion suivi par une classification floue et évaluation de la partition se répète jusqu'au à ce que le nombre des clusters soit égale à la valeur de C_{min}.

III.2.1.1.L'initialisation des Centres

L'algorithme FCM nécessite d'une part une étape de spécification de nombre des clusters, et une étape d'initialisation de leurs centres d'une autre part, cette dernière est obtenue à partir de l'extraction des différentes couleurs du pixel pour calculer la distance de chaque couleur par rapport aux autres, la couleur qui a une distance maximale est inséré comme centre, parmi les couleurs restantes ont choisi la distance maximal par rapport au centre inséré, ainsi obtenir que des couleurs bien séparé

III.2.1.2.La Classification par l'Algorithme FCM (Fuzzy C-Means)

Cette étape consiste à calculer pour chaque pixel de l'image ses degrés d'appartenance aux différentes classes, et les nouveaux centres des qui seront les entrées du FCM, après la convergence une évaluation de la partition floue obtenue est faite, ce processus se répète jusqu'au à ce que le nombre des clusters soit égale à la valeur de C_{min}.

III.2.1.3.L'étiquetage

Après la sélection de la meilleure partition. On doit appliquer l'étape de l'étiquetage sur celle-ci, sous cette étape on affecter chaque pixel à sa classe correspondante selon son degré d'appartenance maximal

III.3.La Conception détaillée

Grace à la conception détaillée on obtient une description des structures de données utilisées et leur communication entre elles pour parvenir en finale à une image classée. On peut résumer notre conception détaillée sous le schéma (Figure III-2)

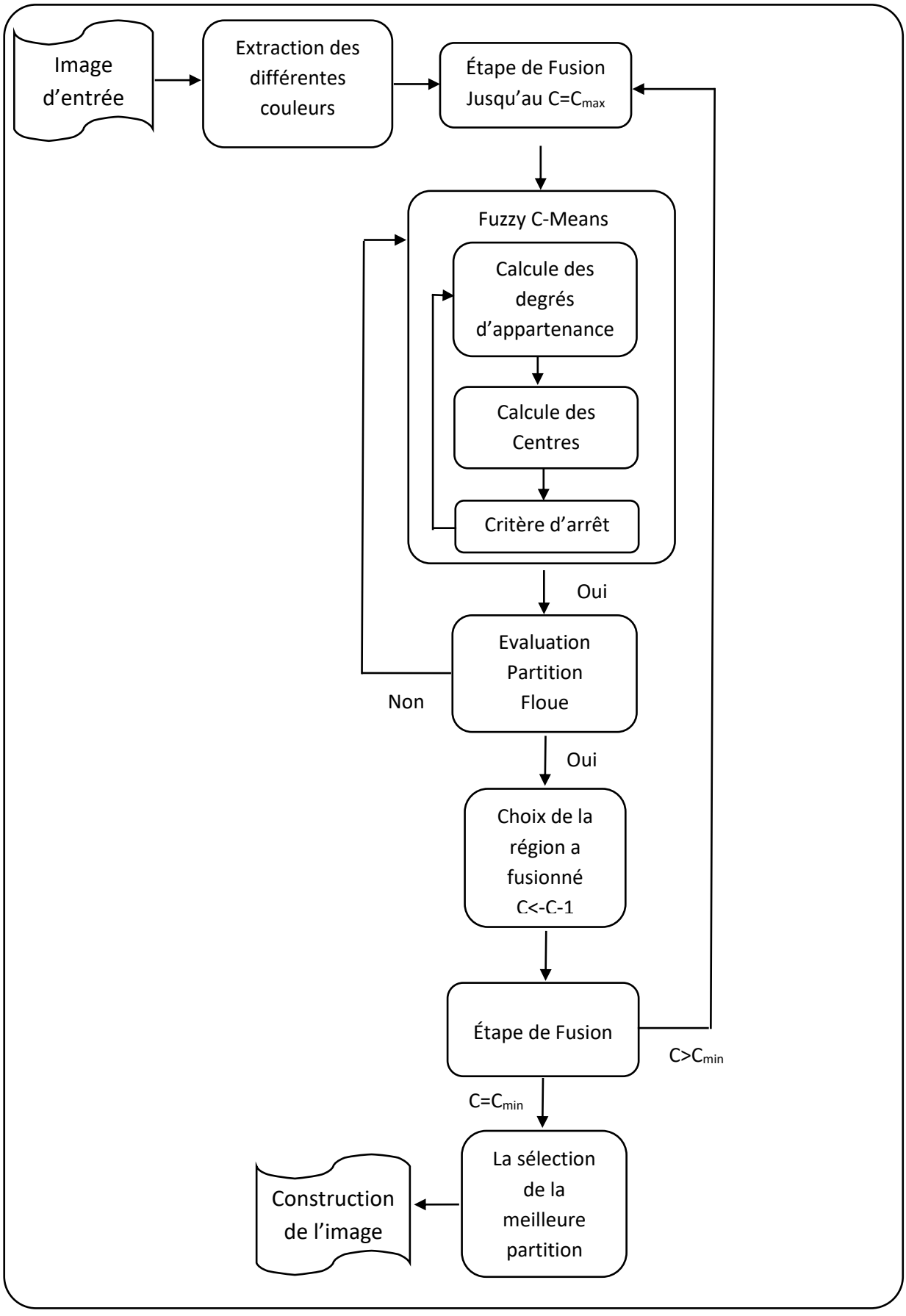

Figure III-2 - Architecture détaillé du Système

III.3.1.Le module d'entrée

Notre application accepte comme une entrée une image numérique. Afin de déterminé le nombre de classes, celle-ci a pour structure de donnée une matrice de pixels repérés par leur cordonnées(x, y) si il s'agit d'une image couleur, un pixel est codé par trois composantes (r, g, b) (chacun compris au sens large entre 0 et 255), représentant respectivement les "doses" de rouge, vert et bleu qui caractérisent la couleur du pixel.

III.3.2.La classification

La méthode utilisée pour la classification d'image dans ce travail est une méthode de classification floue (FCM) combiné avec la stratégie d'énumération, cette dernière est décrite par les étapes suivantes :

III.3.3.L'initialisation des centres

Avant d'appliquer l'algorithme FCM il est nécessaire d'initialiser les centres, Pour cela on doit passer par une étape d'extraction des différents couleurs concernant l'image couleurs, l'initialisation se fait par le calcul de la distance de chaque couleur par rapport aux autres, la couleur qui a une distance maximale est supposé un centre, parmi les couleurs restantes ont choisi la distance maximal par rapport au centre inséré selon un seuil prédéfinie, pour le nombre de centre prédéfinie ce processus ce répète jusqu'au parcours de tous les couleurs de l'image et atteindre et ainsi obtenir des couleurs bien séparé

La structure utilisée pour initialisé les centres est une matrice 2D **clusterCenters [numCenters][numBands],** tel que **numCenters** est le nombre des centres, **numBands** est le nombre des bandes ({ rouge (R), vert (G), bleu (B)}).

III.3.4.L'application de l'algorithme de Fusion

L'étape de fusion est définie par le calcul du fitness de chaque cluster, celle-ci est représentée par la variation au sien du même cluster sur la séparation par rapport aux autres, le cluster de fitness maximal sera le cluster a éliminé ce qui a pour résultat la décrémentation du nombre de cluster ce processus est définie en détail par les étapes suivantes :

- **Fonction Fitness**

Après l'étape d'initialisation des centres et ainsi obtenir les nouveau centres, viens l'étape du calcul du fitness pour chaque cluster, celle-ci est structuré sous forme de tableau **fitness[]** ou elle contient les clusters initiaux à éliminer par la suite

$$fitness(cluster\ i) = \frac{variation(cluster_j)}{séparation\,(cluster_j)} \qquad \text{(III-1)}$$

La variation est définie par la somme des distances entre chaque point appartenant au cluster et le centre sur le nombre de celle-ci

$$Variation(cluster_j) = \sum_{p_i \in cluster_j} \frac{\|p_i - c_j\|}{NC_j} \qquad \text{(III-2)}$$

La Séparation quand a elle est définie par la somme des distances par rapport les autres clusters

$$Séparation\,(cluster_j) = \sum_{i=1, i\neq j}^{C} ||c_j - c_i|| \qquad \text{(III-3)}$$

- **Choix du cluster a éliminé**

Après le calcul du fitness de chaque cluster et afin de réduire ce nombre on doit éliminer le cluster en fonction de sa valeur de fitness, A chaque fois on choisit la fitness maximal de chaque cluster jusqu'à C_{min}

- **L'affectation des points du cluster éliminé**

Pour cela on calcule pour chaque cluster restant la somme de l'ensemble des points de l'image appartenant au même cluster sur le nombre des points de celui-ci, et ainsi repartitionné les points sur les clusters restants

III.3.5.Application de l'algorithme FCM

L'algorithme FCM nécessite une étape de spécification des nombre des classes, et une étape d'initialisation de leurs centres

III.3.5.1.Calcule des degrés d'appartenance

Dans cette étape on cherche à calculer les degrés d'appartenance de notre programme, la structure de données utilisées pour stocker les degrés d'appartenance est une matrice 2D **memberships [width*height][numClusters],** tel que **width*height** est égale à la taille de l'image d'entrée c-à-d le nombre de pixels, et **numClusters** est le nombre des classes.

Pendant cette étape on doit respecter les conditions (1) (2), autrement dit, le degré d'appartenance d'une certaine couleur aux différentes classes se varie entre 0 (degré d'appartenance minimal à une classe) et 1 (degré d'appartenance maximal à une classe). De plus, la somme des différents degrés d'appartenance d'une seule couleur doit être égale à 1.

III.3.5.2.Le Calcule Fonction Objective

La minimisation de la fonction objective $J(C, U, X)$ peut être amenée par un processus itératif dans lequel la mise à jour du degré d'appartenance u_{ij} et les centres de classes c_i sont faits pour chaque itération de la classification

III.3.5.3.Le Calcule des centres

La minimisation de la fonction objective nécessite une mise à jour itérative des centres, Et pour garder les centres calculés on a utilisé une matrice 2D **clusterCenters [numClusters] [numBands]**, où les lignes représentent les nombres des classes et les colonnes représentent le nombre des bandes.

III.3.5.4.Le critère d'arrêt

Le FCM converge lorsque les centres des classes n'évoluent pas entre deux itérations successives, c'est-à-dire lorsque les valeurs de tous les centres entre deux itérations successives ne dépassent un certain seuil.

III.3.6.Evaluation de la meilleure partition

Pour réaliser le partitionnement approprié (maximiser la similarité au sein les classes elles-mêmes et minimiser la similarité entre les classes) la valeur de l'indice de XB doit être plus faible c'est à dire la distance minimale entre tous les centres est élevée. [16]

III.3.7.L'étiquetage

L'objectif de cette étape est la construction de l'image finale à partir de la matrice des degrés d'appartenance obtenu après la convergence de l'algorithme de classification, et pour réaliser cette étape on affecte chaque couleur (les pixels caractérisent par cette couleur) à sa propre classe (la classe dont le degré d'appartenance est le plus élevé).

```
Procédure Fusion ( )
Var numBunds, weight, height, epsilon, fizziness, index: entier;
    distance,minDistance,interDistance : réel ;
    memberShips[weight*height][numClusters], clusterCenters[numClusters][numBunds],
    eliminatedCluster[centers], aPixel [numBands] : réel ;
Début
  Lire_Image(Image) ;
  Image [][] ←getImageInMatrice() ;
  Colors [][] ← getColors() ;
  Centers [][] ← initial_Centers() ;
  Etiquettes[] ←Etiquetage() ; //Etiquetage des points de l'image avec les centres initiaux
  Pour (i ←0 à centers[i]) Faire
    Nb ← 0;
      Pour (j ←0 à etiquettes[j]) Faire
        Si (etiquettes[j]=i) Alors
          Pour (int b=0;b<numBands;b++) Faire
            centers[i][b] ← centers[i][b]+image[j][b];
            nb←nb+1;
          Fin Pour
        Fin Si
      Fin Pour
      Pour (b ← 0 à numBands) Faire
        centers[i][b] ← centers[i][b]/nb;
      Fin Pour
  Fin Pour
  Pour (i ← 0 à centers[i]) Faire
    Pour (j ← 0 à etiquettes[j]) Faire
      Si (etiquettes[j]=i) Alors
        freq[i] ← freq[i] +1;
      Fin Si
    Fin Pour
  Fin Pour
 Tant que (nbCluster≠centers) Faire
    Pour (i ← centers[i]) Faire
      nb ← 0;
     Pour (j ← etiquettes [j]) Faire
        Si (etiquettes[j]==i)
          Pour (int b=0;b<numBands;b++) Faire
            fitness[i] ← fitness[i]+this.calcDistance(centers[i], image[j]);
            nb← nb+1;
          Fin Pour
        Fin Si
      Pour (b ← 0 à numBands) Faire
          fitness[i] ← fitness[i]/nb;
      Fin Pour
    Fin Pour
  Fin Pour
```

```
Si (nbClusters<=Cmax et nbCluster>=Cmin) Alors
  lastJ ← calcule_Fonction_Objective() ;
  Tant que ((LastJ-j) < epsilon ) Faire
    Pour (iteration ← 1 à maxIteration) Faire
        Calculate_Membership () ;
        clusterCenters() ;
        j ← calculate_fonction_objectif() ;
        lastJ ← j ;
    Fin Pour
  Fin Tant que
  Si (nbClusters = Cmax) Alors
       min ← indice_Xie_Beni();
       nbCmin = 12;
  Sinon
    double r1 = indice_Xie_Beni();
    Si (r1 < min) Alors
      min ← r1;
      nbCmin ← nbClusters;
    Fin Si
  Fin Si
  hasFinished ← Vrai ;
  Position ← getSize () ;
  Fin Si
 Fin Tant que
Fin
```

```
Fonction calcDistance (a1 [ ], a2 [ ] : réel) : réel ;
Var Distance : réel ;
        e: entier;
Début
    Distance ← 0;
    Pour (e ←1 à a1.length) Faire
      Distance ← Distance + (a1[e]-a2[e])*(a1[e]-a2[e]) ;
    Fin Pour ;
    Retourne (carré [distance]) ;
Fin.
```

```
Procedure Fitness() ;
Var Nb, nbCluster, InterFitness, MaxFitness, indiceFitness, i, j, k : entier ;
    Fitness[centers], EliminatedCluster[centers] : entier ;

Debut
  Pour (i←0 à Centers[i]) Faire
   Pour (j ←0 à Etiquettes[j]) Faire
    Si (Etiquettes[j]=i) Alors
      Fitness[i] ← Fitness[i]+calcDistance(center[i],image[j]) ;
      nb←nb+1 ;
     Fin Si
     Fitness[i] ←Fitness[i]/nb ;
    Fin Pour
  Fin Pour
  Pour i ← (0 à Fitness[i]) Faire
    Si (eliminatedClusters[i] ≠ 0) et (fitness[i]>maxFitness) Faire
       maxFitness←fitness[i];
       indiceFitness←i;
    Fin Si
    eliminatedClusters[indiceFitness] ← 0;
  Fin Pour
  Tant que (k<eliminatedClusters.length) et (bol=faux) Faire
    Si (k ≠ indiceFitness) et (eliminatedClusters[k] ≠ 0)
     Bo ← true;
     indiceInitial ← k;
    Fin Si
     k ← k+1;
  Fin Tant que
 Pour (j ←0 à centers[j]) Faire
    Si (j!=indiceFitness && eliminatedClusters[j]!=0) Alors
      Pour (int i1=0;i1<centers.length;i1++) Faire
       Si (i1!=indiceFitness && eliminatedClusters[i1]!=0) Alors
        interDistance ← interDistance+calcDistance(centers[j], centers[i1]);
        distance ← this.calcDistance(image[i],centers[j]);
         Si (distance<minDistance) Alors
          minDistance ← distance;
          indCluster ← j;
         Fin Si
        Fin Si
      Fin Pour
     Fin Si
  Fin Pour
Fin
```

```
    Procédure Initial_centres ( )
Var indiceMax, nbCenters,ind, frequence [colors.length], marquage[colors.length],
indiceCenters [colors.length], centerRegions[numCluster][numBands] : entier;
    max, interDistance, maxDistance, TotDistance [colors.length] : réel ;
Début
    Pour (i ← 0 à colors[i]) Faire
      Pour (j ← 0 à colors[j]) Faire
        SI (i≠j) Alors
          TotDistance [i] ← calcDistance (colors[i],colors[j]);
        Fin Si;
      Fin Pour ;
    Fin Pour ;
    max ← TotDistances [0];
    indiceMax ← 0 ;
    Pour (i ← 0 à TotDistance) Faire
      SI (TotDistances[i]>max) Alors
        Max ← TotDistances[i];
        indiceMax ← i;
      Fin Si;
    Fin Pour;
   nbCenters ← 1;
   maxDistance, ind, interDistance ← 0;
  Tantque (nbCenters < 20) Faire
    Pour (i ← 0 à colors[i]) Faire
    SI (marquage[i]=0) Alors
       interDistance ← calcDistance (colors[i], centerRegions[nbCenters-1]);
      SI (interDistance>maxDistance) Alors
        Pour (j ← 0 à nbCenters) Faire
          interDistance ← interDistance+this.calcDistance (colors[i],centerRegions[j]) ;
        Fin Pour;
          SI (interDistance>maxDistance) Alors
            maxDistance ← interDistance;
            ind ← i;
          Fin Si;
        Sinon
          Marquage [i] ← 1;
        Fin Si;
      Fin Si;
    Fin Pour;
    Pour (b ← 0 à numBands) Faire
      centerRegions [nbCenters][b] ← colors[ind][b];
    Fin Pour;
  Fin Tant que;
      Retourne (CentreRegions) ;
  Fin ;
```

```
Fonction get.Colors (matrice [][] : réel) :réel ;
Var length=height*width, size=0, i , j, k, indice : entier
Boolean bol = vrai ;
Début
 mat [][] = getImageInMatrice();
 mat2 [][] = [length][numBands];
 Pour (k←0 à numBand) Faire mat2[0][k] ← mat[0][k];
  Tant que (i<length) Faire
   Tant que (j<i) et (bol=vrai) Faire
     Si (mat[i][0]=mat[j][0]) et (mat[i][1]=mat[j][1]) et (mat[i][2]=mat[j][2]) Alors bol←faux;
     Sinon     j←j+1;
   Fin tant que
   Si (bol=Vrai) Alors
    Pour (k←0 à numBands) Faire
      mat2[indice][k] ← mat[i][k];
      indice←indice+1;
    Fin Pour
   Fin Si
  Fin Tant que
    j=0;
    bol=true;
    i←i+1;
 Fin Pour
     Retourner (mat2);
Fin
```

```
Procedure Etiquettage()
Var : Etiquettes[],centres[],image[] : entier ;
      Indice,Distance,MinDistance : entier ;
Debut
Initial_centers() ;
 Pour (i←0 à Image[i]) Faire
  Pour (j←0 à centres[j]) Faire
   Distance←Calcul_Distance(Center[i],Image[j]) ;
   Si(Distance<MinDistance) Alors
    MinDistance←Distance ;
    indice←j ;
   Fin Si
  Fin Pour
 Etiquettes[i] ←indice ;
 Fin Pour
Retourne(Etiquettes) ;
Fin
```

```
Procédure Calcule_degres_appartenances ( )
Var w, b, c, ck, index: entier;
      Top, Sum, Distance, aPixel [numBands] : réel ;
Début
   Pour (c ← 1 à numClusters) Faire
     Pour (w ← 1 à (width*height)) Faire
       Index ← w*numBands;
       Pour (b ← 1 à numBands) Faire
         aPixel [b] ←InputData [index + b];
       Fin Pour ;
       Top ←calcDistance (a Pixel, clusterCenters[c]);
       Sum ← 0 ;
       Pour (ck ← 1 à numClusters) Faire
         Distance ← Distance + calcDistance (a Pixel, clusterCenters [ck]);
         Sum ← Sum + [(top/Distance) puissance (2/ (fuzziness-1))];
       Fin Pour;
       Membership [w] [c] ←1/ Sum;
     Fin Pour ;
   Fin Pour ;
Fin.
```

```
Procédure Calcule_Centres ( )
Var w, b, c, index : entier ;
      Top, Bottom : réel ;
Début
   Pour (b ← 0 à numBands) Faire
     Pour (c ← 0 à numClusters) Faire
       Top ←0 ;
       Bottom ← 0;
       Pour (w ← 0 à (width*height)) Faire
         Index ← w*numBands;
         Top ← Top + [membership[w] [c] puissance fuzziness]*InputData [index+b];
         Bottom ← Bottom + [membership[w] [c] puissance fuzziness];
       Fin pour ;
       ClusterCenters [c] [b] ← Top/ Bottom;
     Fin Pour ;
   Fin Pour ;
Fin.
```

```
Fonction calcule_Fonction_Objective ( ) : entier ;
Var j, w, b, c, index : entier ;
      DistancePixelToCluster, aPixel [numBands] : réel ;
Début
    Pour (w ← 1 à (width*height)) Faire
        Pour (c ← 1 à numClusters) Faire
          Index ← w*numBands ;
          Pour (b ← 1 à numBands) Faire
            aPixel [b] ← InputData [index + b];
          Fin Pour ;
          DistancePixelToCluster ← calcDistance (a Pixel, clusterCenters[c]) ;
          j ←j+ DistancePixelToCluster*[membership[w] [c] puissance fuzziness];
        Fin Pour ;
    Fin Pour ;
     Retourne (j) ;
Fin.
```

```
   Fonction Indice_Xie_Beni ( ) : entier ;
Var w, b, c, cs, c1, c2, index : entier ;
      DistancePixelToCluster, minDist, distance : réel ;
Début
  Pour (w ← 1 à (width*height]) Faire
    Index ← w*numBands;
    Pour (b ← 1 à numBands) Faire
      aPixel [b] ← InputData [index + b];
    Fin Pour ;
    Pour (c ← 1 à numClusters) Faire
      DistancePixelToCluster = calcSquaredDistance (a Pixel, clusterCenters[c]);
      cs ← cs + memberships[w][c]*memberships[w][c]*DistancePixelToCluster
            *DistancePixelToCluster;
    Fin Pour ;
  Fin Pour;
  cs ← cs / (height*width) ;
  minDist ← Float.MAX_VALUE;
  Pour (c1 ← 1 à (numClusters-1)) Faire
    Pour (c2 ← c1+1 à numClusters) Faire
      Distance ← calcSquaredDistance(clusterCenters [c1],clusterCenters [c2]);
      MinDist ← minimum (minDist, distance) ;
    Fin Pour ;
  Fin Pour ;
  cs ← cs / (minDist*minDist) ;
  Retourne (cs) ;
Fin.
```

III.4.Conclusion

Dans ce chapitre, nous avons présenté les différentes étapes du fonctionnement de notre système de classification des images en passons de la conception général a détaillé en mentionnant à chaque étape la structure et algorithme utilisé. Notre choix s'est porté sur la classification FCM basé sur les algorithmes génétiques. Le chapitre suivant est consacrée à l'implémentation de notre système.

Chapitre IV :

L'Implémentation

Chapitre IV : L'Implémentation du Système

IV.1.Introduction

Dans ce dernier chapitre et après l'aperçu théorique des chapitres précédents, nous présenterons le côté pratique de l'application. Le but est la réalisation d'un système qui détermine le nombre de classes des images couleurs. Nous commençons par le choix de l'environnement de travail en passons par le langage et le logiciel de codage la bibliothèque intégrée ainsi qu'une brève présentation de l'application, aussi les fonctionnalités de chaque composant du système, pour finir l'illustration de quelques résultats.

IV.2.L'environnement de travail

Pour que notre travail atteint l'objectif qu'on visait, on a pris l'initiative d'exploiter et d'implémenter notre algorithme sur : Windows 10 (Figure IV-1). Ce choix se traduit par la nouveauté et l'efficacité de cet environnement en ce qui concerne la structure d'interaction événementielle qu'elle dispose pour communiquer avec des applications actives, ainsi que les ressources de la machines qu'il offre aux différentes applications, enfin, son système d'allocation de mémoire qui est un des meilleurs présents dans ce domaine.

Figure IV-1 - Le Logo du Système d'exploitation

IV.3.Le langage de codage

Notre application a été codée en sa globalité par le langage JAVA (Figure IV-2) qui est un langage de programmation informatique orienté objet à travers la plateforme NetBeans

Figure IV-2 - Le Logo de Java

Ce choix repose sur le fait qu'il est :

- Populaire : en particulier en entreprise, le langage Java est un investissement pour celui qui l'apprend,
- Riche : il existe de nombreuses bibliothèques dans tous les domaines, celles-ci ont l'avantage considérable d'être standardisées,
- International : les programmes Java supportent les normes internationales Unicode pour le codage des caractères de toutes les langues,
- Propre : il préserve un typage assez fort des données en présence, les Operations arithmétiques sont très bien spécifiées.

IV.4.Logiciel de codage

Les environnements de développement intégrés (EDI), sont des logiciels regroupant un ensemble d'outils nécessaires au développement logiciel dans un (ou plusieurs) langage(s) de programmation. Parmi tous les environnements de développement existant notre choix a porté sur NetBeans (Figure IV-3) à la faveur de sa rapidité à mettre en place une application qui est l'un des points forts de ce dernier

Figure IV-3- Le Logo de NetBeans

Une boîte à outils qui permet de gagner beaucoup de temps en développement et en effort. Un ensemble de Standards pour rehausser et renforcer la consistance et l'interopérabilité entre les applications et les systèmes d'exploitation.

La Figure IV-4 suivant présente l'interface de NetBeans 6.8 que nous avons utilisé pendant notre implémentation.

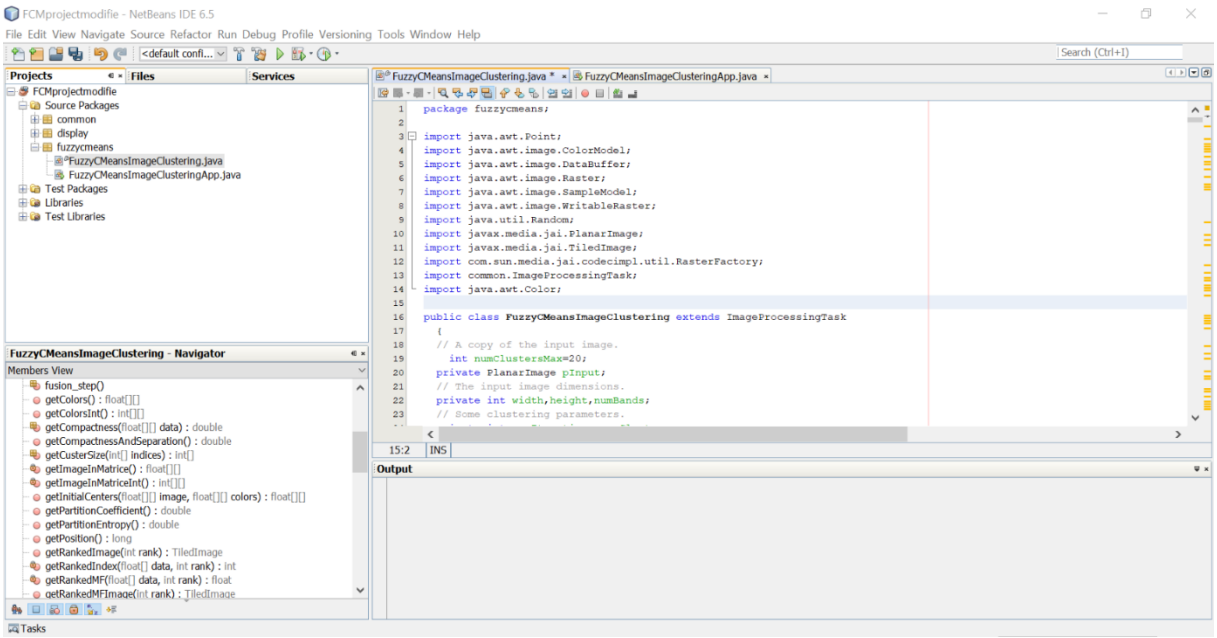

Figure IV-4- L'interface de l'environnement.

IV.5.Bibliothèque Utilisée

L'API Java Advanced Imaging (JAI) implémente un ensemble de fonctionnalités qui étend encore la plate-forme Java en permettant l'intégration d'un traitement d'image sophistiqué.

JAI est destiné à répondre aux besoins de toutes les applications d'imagerie. L'API est très extensible, ce qui permet d'ajouter de nouvelles opérations de traitement d'image de manière à en faire partie intégrante. Ainsi, JAI profite à pratiquement tous les développeurs Java qui veulent intégrer l'imagerie dans leurs applets et applications.

Parmi les bibliothèques que nous avons eues recourt durant l'implémentation :

- jai-1_1_3-lib-windows-i586
- jai-1_1_3-lib-windows-i586-jdk
- jai-1_1_3-lib-windows-i586-jre

IV.6.Caractéristique de la Machine

L'implémentation du projet est réalisé par sur un ordinateur portable qui se caractérisé techniquement par :

- Windows 10 Pro 64-bit
- Intel Core i7-3612QM CPU @2.10 8,0GB Ram, Intel HD Graphics 4000

IV.7.Présentation de l'Application

Dans notre application nous offrons à l'utilisateur une interface graphique qui facilite la manipulation des paramètres pour la rendre plus compréhensible.

IV.7.1.Fenêtre d'Accueil

Cette fenêtre (Figure IV-5) présente l'accueil de l'application qui contient des informations et le bouton de démarrage

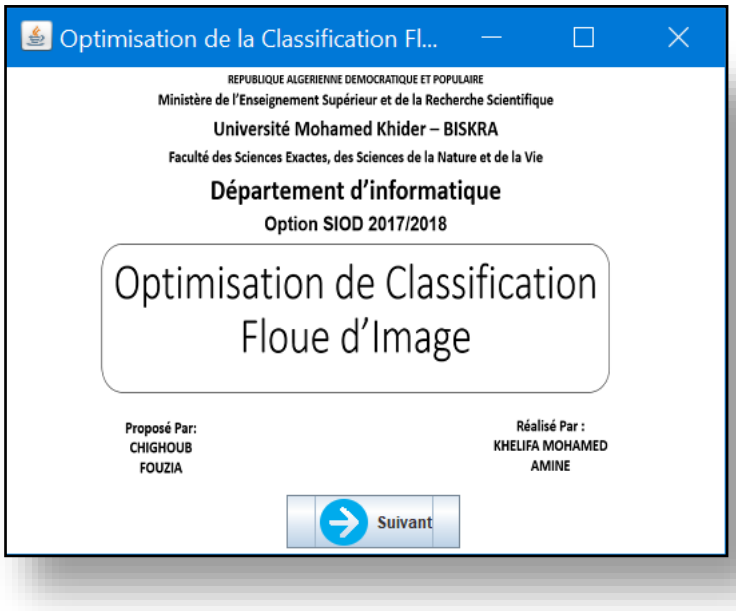

Figure IV-5 - La Fenêtre d'accueil

IV.7.2.Interface et composants

La figure ci-dessous (Figure IV-6) montre la fenêtre principale et représente les éléments essentiels de notre application.

Figure IV-6- Interface de l'application.

En cliquant sur « Fichier » un menu déroulant apparaît vous permettant d'effectuer les fonctions suivantes (Figure IV-7) :

Ouvrir : L'utilisateur en cliquant dessus peut choisir une autre image.

Figure IV-7- Interface de l'application (Ouvrir)

Celle-ci n'affiche que les fichiers Images (Figure IV-8) :

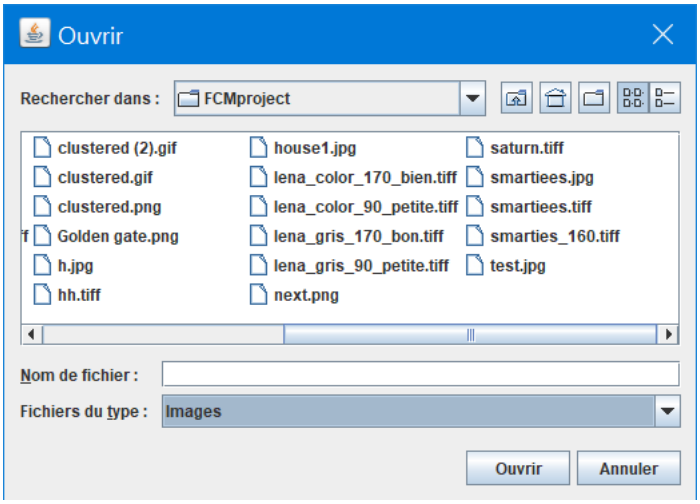

Figure IV-8 - Fenêtre pour parcourir d'autres images

Enregistrer : Pour sauvegarder l'image après modification (Figure IV-9)

Figure IV-9 - Interface de l'application (Enregistrer).

Quitter : Pour quitter l'application (Figure IV-10)

Figure IV-10- Interface de l'application (Quitter).

Ensuite une Fenêtre s'affichera pour confirmer la sortie du logiciel (Figure IV-11)

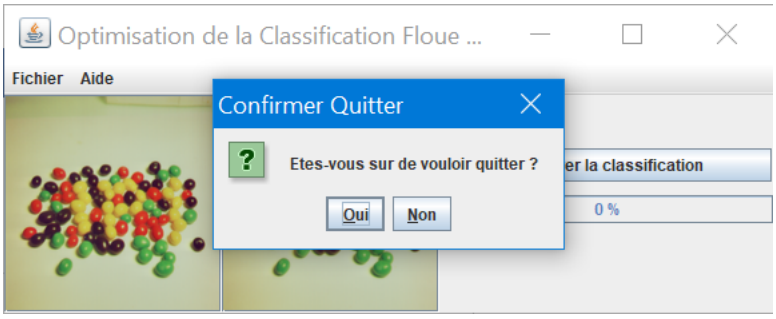

Figure IV-11 – Fenêtre de confirmation de Quitter

Pour finir une fenêtre d'appréciation s'affichera à la fin (Figure IV-12)

Figure IV-12 – Fenêtre d'appréciation

Ici l'interface principale du logiciel et le rôle de chacune (Figure IV-13)

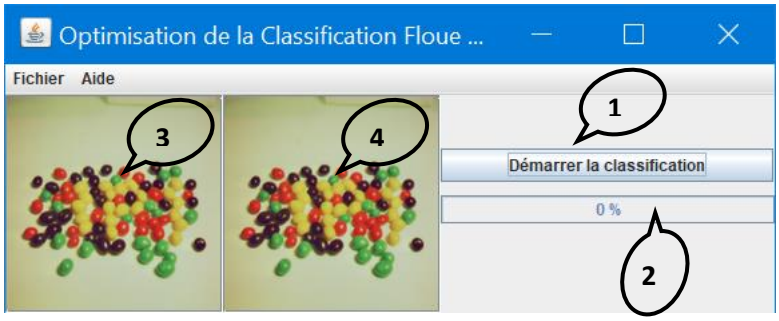

Figure IV-13 - Les composants d'interface de l'application.

1) **Démarrer la classification :** Le bouton de démarrage de la méthode de classification.

2) **La barre de changement :** Cette barre indique la progression de l'avancement de la classification.

3) **L'image Initiale** avant classification.

4) **L'image résultat** après classification.

Les figures suivantes présentes quelques résultats de notre application de classification sur des images avec de nombre des classes différents.

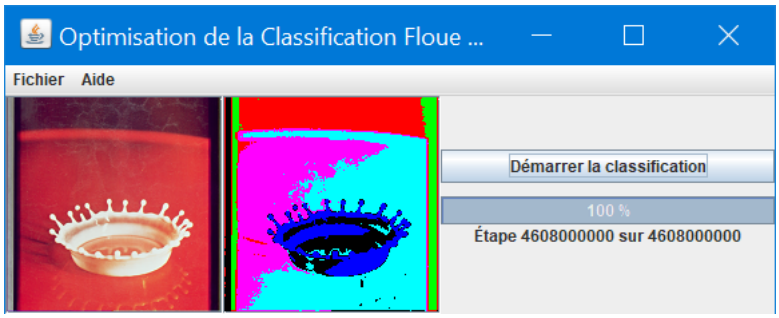

Figure IV-14 – La classification d'une chute d'une goutte (6 classes)

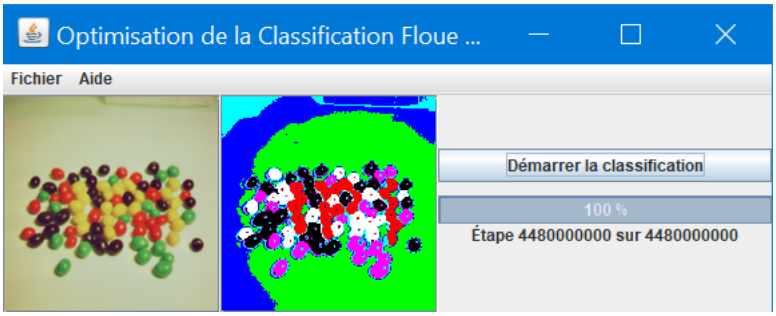

Figure IV-15 - La classification de l'image Smarties (7 Classes)

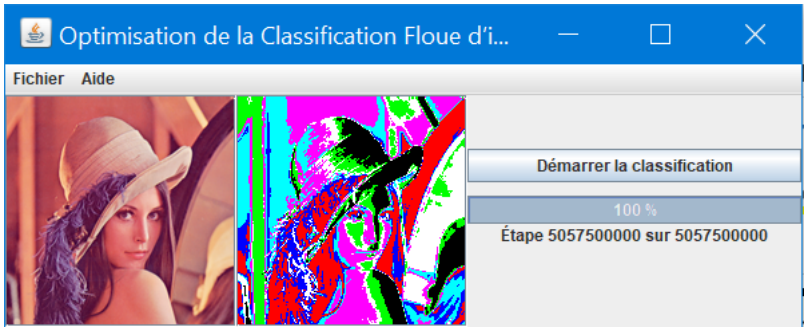

Figure IV-16- La classification de l'image Lena (7 Classes).

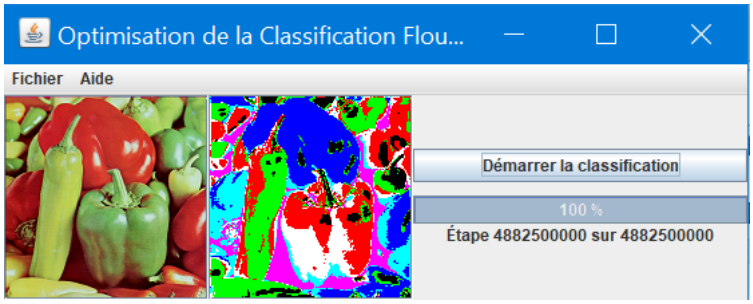

Figure IV-17- La classification de l'image Capsicum (7 Classes).

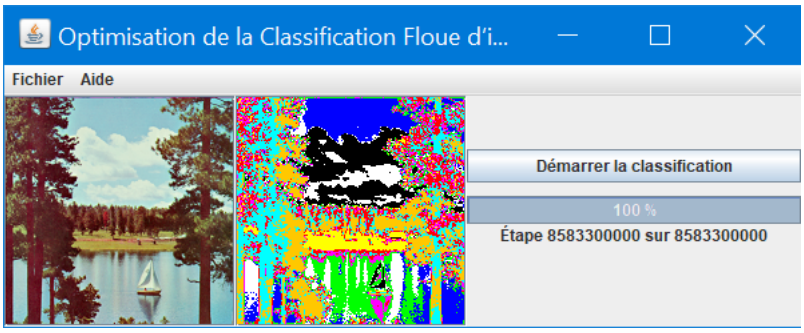

Figure IV-18 - La classification de l'image paysage (8 Classes)

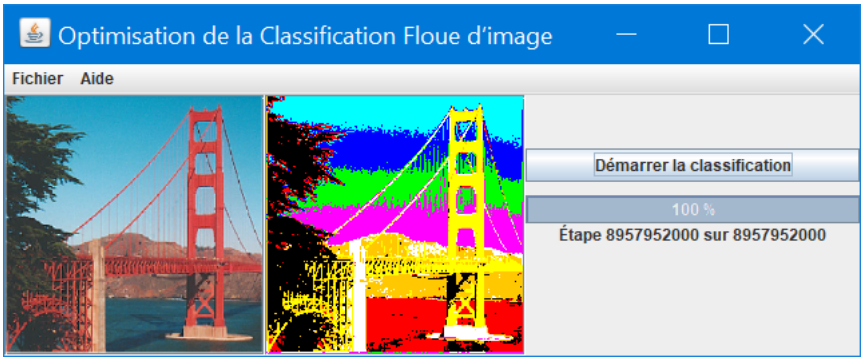

Figure IV-19 - La classification de l'image pond (8 Classes)

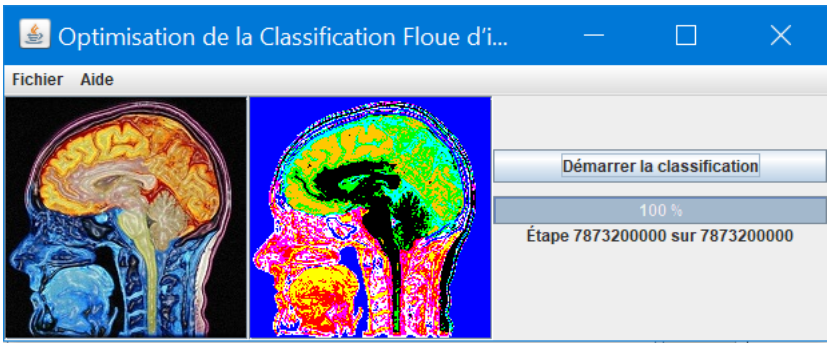

Figure IV-20 - La classification de l'image crane (8 classes)

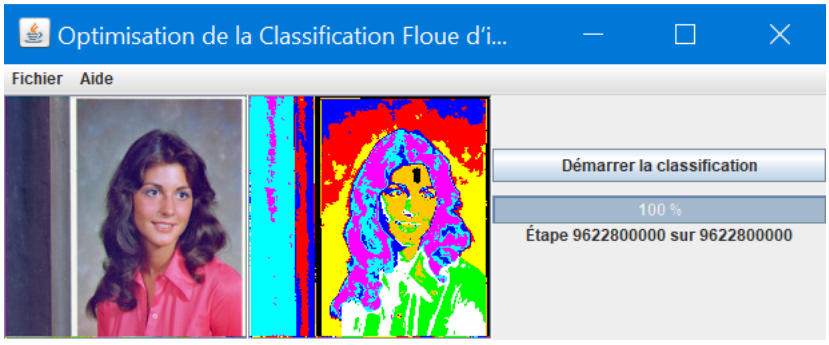

Figure IV-21 - La classification de l'image portrait (8 classes)

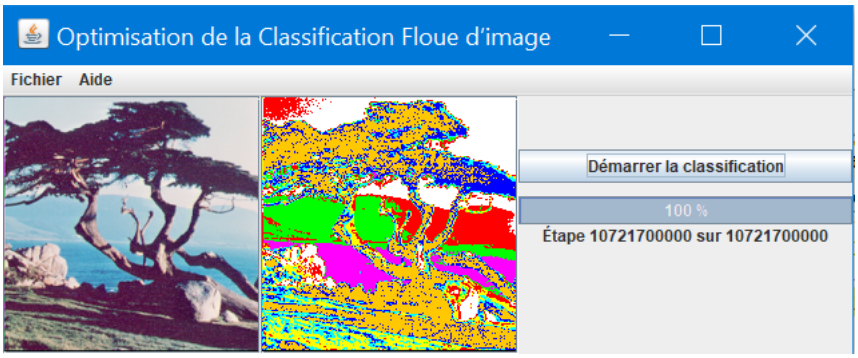

Figure IV-22 - La classification de l'image arbre (9 Classes)

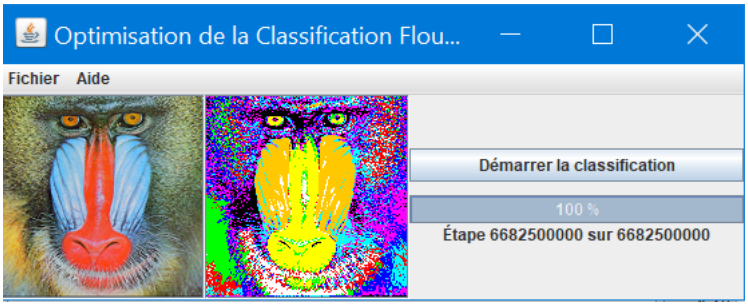

Figure IV-23 - La classification de l'image singe (10 Classes)

IV.8.Conclusion

Dans ce chapitre on a vu la réalisation de notre conception, alors on a présenté tous les composants qui produisent cette application, les différents cas pour l'utilisation de notre système et nous avons présenté également quelques résultats de notre méthode de classification.

Conclusion Générale

Conclusion Générale

La segmentation d'images est une étape cruciale dans tout processus d'analyse d'images. Elle consiste à préparer l'image afin de la rendre plus exploitable par un processus automatique telle que l'interprétation.

De manière non exhaustive, nous avons présenté dans le premier chapitre les différentes techniques de la segmentation. Il existe deux grandes approches purement locales. L'approche contour consiste à localiser les frontières des régions, elle est basée sur la notion de dissimilarité. Parmi ces point fort : sa simplicité et sa rapidité mais elle donne parfois des contours ouverts. L'approche région consiste à réunir les pixels connexes dans une région homogène, elle est basée sur la notion de similarité. Elle est simple et rapide mais l'utilisation uniquement des informations locales donne parfois de mauvais résultats (sous-segmentation, sur-segmentation). La coopération de ces deux approches qui sont par nature dual améliore les résultats de la segmentation. Nous nous sommes particulièrement intéressés à l'approche région dans laquelle les méthodes non supervisé basées sur la classification et plus précisément la classification floue (FCM) qui a attiré notre attention.

L'algorithme FCM est une méthode de classification non supervisée très populaire et très performante par rapport aux autres Il est abondamment utilisé pour la segmentation d'images, Malgré cela, l'algorithme FCM souffre de certains inconvénients parmi lequel la problématique de la détermination des nombres de classes dans l'algorithme FCM que nous avons essayé d'abordé par le biais d'intégration des techniques d'énumération

Notre algorithme de classification a été testé sur des images diverses, Nous avons pu constater à travers nos différents tests que les résultats obtenus montrent que la méthode de classification floue par l'intégration des algorithmes d'énumération apporte de nettes améliorations par rapport à l'algorithme FCM classique.

Nous avons implémenté une application qui permet à l'utilisateur de choisir une image pour classer ces éléments. Et offert une interface interactive qui facilite l'utilisation de ces différents composants.

Conclusion Générale

Ce travail a permis de mettre en pratique de nombreuses connaissances dans le domaine de la classification d'image et de les approfondir. Divers perceptives peuvent être envisagées, tout d'abord la méthode de classification floue basée sur les méthode d'énumération implémentée en ce travail nécessite l'intégration de l'information spatiale afin d'améliorer la robustesse du résultat de la classification, et on l'intégration d'un module de prétraitement afin de réduire l'effet du bruit sur l'image, et on conclue cette partie par la dernière perspective qui consiste à utiliser des informations à priori pour établir l'étape d'initialisation des paramètres de méthode (les centres, la matrice des degrés d'appartenance) afin d'accélérer la convergence du processus de algorithme.

La Bibliographies

Bibliographie

1. **Elberkanie M., Belmeskine R., Drief S., Segmentation d'une image split & merge, Mini Projet Master, université Mohammed Premier Oujda, 2010/2011.**

2. **Haralick, R. M.Image segmentation techniques. L. G, 1985.**

3. **Jean Jaques Rousselle. Les contours actifs, methode de segmentation. Application a l'imagerie medicale. These Universite Fracois Rabelais de Tours, 09 juillet 2003.**

4. **LAKHDARI Mohamed .Segmentation d'images par contour actif en appliquant les algorithmes.**

5. **Acharya, T. A."Image Processing, Principles and Applications" chapitre 7,2005.**

6. **M Sandeli, traitement d'images par des approches bio-inspirées application à la segmentation d'images, université constantine 2. 2014.**

7. **M. Melliani, segmentation d'image par cooperation regions-contours, magistère en informatique, ecole national supérieur d'informatique, 2012.**

8. **C.Houassine, segmentation d'images par une approche biomimétique hybride. université universite m'hamed bougara- boumerdes. 2012.**

9. **Amir Samir. Evaluation de la segmentation d'image sans référence étude et implémentation, Université de Blida, 2005.**

10. **R Haroun. Segmentation des tissus cérébraux sur des images par résonance magnétique. Master's thesis, Université des sciences et de la technologie Houari Boumediène, 2005.**

11. **Imane Sebari et Dong-Chenhe. Les approches de segmentation d'image par coopération régions-contours, Université de Sherbrooke, 10 Avril 2007.**

12. **LEONARDI, Valentin, modelisation dynamique et suivi de tumeur dans le volume renal these de doctorat, 13 Novembre 2014.**

13. **ASSAS Ouarda, Classification floue des images, Thèse de Doctorat Université de Batna, 2013.**

14. **Sbili lila, Segmentation d'images par classification floue, Mémoire de magister,Université Mouloud MAMMERI de Tizi-Ouzou, 08 Mars 2015.**

15. **Eziddin Wael, Segmentation itérative d'images par propagation de connaissances dans le domaine possibiliste : application à la détection des tumeurs en imagerie mammographique.**

16. **Ujjwal Maulik, Sanghamitra Bandyopadhyay, Fuzzy Partitioning Using Real Coded Variable Length Genetic Algorithm for Pixel Classiffcation, May 2003, pp 1075 - 1081.**

17. **Haojun Suna, Shengrui Wang, Qingshan Jiang, FCM-Based Model Selection Algorithms for Determining the Number of Clusters, Pattern Recognition 37, pp 2027 – 2037, 17 June 2004.**

18. **Jürgen Beringer, Eyke Hüllermeier, Fakultät für Informatik, Adaptive Optimization of the Number of Clusters in Fuzzy Clustering, Fuzzy Systems Conference, 2007, pp 1-6.**

19. **Zhehui Liang, Pingjian Zhang, Juanjuan Zhao, Optimization of the Number of Clusters in Fuzzy Clustering. Computer Design And Applications International Conference, pp 580-584, 27 June 2010.**

20. **Bensaid, Amine M, Validity -Guided (Re)Clustering with Applications to Image Segmentation, transactions on fuzzy systems,vol. VOL. 4, NO. 2, MAY 1996. pages 115, 116.**