Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Institut National de formation en Informatique (I.N.I)
Oued-smar Alger

Mémoire de fin d'études

En vue de l'obtention du diplôme d'Ingénieur d'État en Informatique

Option : Systèmes Informatiques

THÈME

Approches de segmentation d'images par méthodes biomimétiques

Réalisé par:

M^r Anis Benyelloul
M^r Said-Eddine Bensalem

Proposé par:

M^r M. Koudil
M^r B. Souici

Promotion: 2006/2007

Remerciements

Nous adressons notre profonde gratitude à l'ensemble du Corps Enseignants de L'INI ayant contribué à notre formation.

Nos vifs remerciements vont à nos Promoteurs Mr Koudil et Mr Souici pour leurs conseils, leurs recommandations et leurs encouragements tout au long de cette année.

Nos vifs remerciements vont également aux enseignants du laboratoire de l'INI, en particulier M^{elle} Benatchba pour son aide et sa constante disponibilté.

Que les membres du Jury trouvent ici le témoignage de notre reconnaissance pour avoir bien voulu juger notre travail.

Nous tenons également à remercier nos familles, et nos amis, pour leurs soutients constants et leurs encouragements.

Résumé

Le travail présenté dans ce mémoire consiste en l'étude de méthodes inspirées de comportements naturels pour la segmentation d'images. Les méthodes choisies pour cette étude sont les deux algorithmes : *Ainet* et *PSO*. *Ainet*(Artificial immune network) est un algorithme inspiré des réactions immunitaires des vertébrés, tandis que PSO (Partical Swarm Optimization) est inspiré du déplacement collectif observé chez les oiseaux migrateurs.

Le problème de la segmentation d'images est considéré comme un problème de partitionnement de données. Dans cette optique, les algorithmes étudiés sont modélisés pour résoudre un problème de classification des pixels d'une image.

L'algorithme *Ainet* utilise une méthode d'apprentissage pour séparer les différentes classes de pixels, tandis que l'algorithme *PSO*, considère la segmentation d'images comme un problème d'optimisation combinatoire, avec comme objectif de trouver une solution *optimale* qui définit l'ensemble des centres des classes minimisant une fonction objectif.

Les méthodes ont été testées sur des images médicales, et comparées avec deux algorithmes classiques de partitionnement : K-means et fuzzy-C-means. Des hybridations de *Ainet* et du *PSO* avec le fuzzy-C-means ont été effectuées, ce qui a permis d'améliorer la qualité de la segmentation, et plus particulièrement les résultats de la méthode *PSO*.

Mots clés : *Segmentation, partitionnement des données, optimisation, algorithmes biomimétiques.*

Abstract

The work described in this report, is the study of methods inspired from natural behaviors for the image segmentation problem. For this study, we have chosen two biomimetic algorithms: *Ainet* and *PSO*. *Ainet* (stands for Artificial immune network), is an algorithm inspired from immune reactions of vertebrates, while the *PSO* (which stands for Partical Swarm Optimisation) is a population-based algorithm inspired by social behavior of bird flocking.

In this report, image segmentation is considered as a data clustering problem. Thus, the studied algorithms are modeled to fit with a problem of pixel classification.

The *Ainet* algorithm uses a learning-based method to separate pixel classes in the image. While *PSO* considers the segmentation as a combinatorial optimization problem, with a goal of finding an *optimal* solution, which represents a set of class centers minimizing a fitness function.

The methods have been tested on medical images, and compared with two well-known clustering algorithms: K-means and fuzzy-C-means. Moreover, the *Ainet* and *PSO* have been hybridized with fuzzy-C-means in order to enhance the quality of segmentation results.

Key words : *Segmentation, data clustering, optimization, biomimetic algorithms.*

Table des matières

Table des figures

Liste des algorithmes

Chapitre 1

Introduction générale

1.1 Introduction au problème posé

Le développement technologique au sens large, a apporté des améliorations considérables dans plusieurs domaines, tels l'industrie, l'agriculture, ou encore la médecine. Un domaine particulièrement intéressant affecté par ce progrès, est l'analyse et la compréhension du monde extérieur, connue sous le nom de *vision artificielle*.

Ce système est défini comme étant une reproduction artificielle du système de vision humain. Ainsi, les différentes étapes du processus naturel sont simulées par ordinateur. Le mécanisme de vision artificielle peut être divisé en trois étapes :

Acquisition de la scène : Il s'agit du processus de production d'images exploitables par ordinateur à partir de l'environnement à étudier.

Prétraitement : le but de cette étape est l'amélioration des résultats obtenus dans l'étape d'acquisition.

Segmentation : Il s'agit du processus de division de l'image en ses différentes parties, dans le but de l'analyser au niveau objets. Le résultat de cette étape affecte fortement les étapes supérieures.

Traitement de haut niveau : Dans cette étape, le traitement consiste en l'analyse et la manipulation des différents objets obtenus dans l'étape de segmentation.

L'étape à laquelle nous nous intéressons particulièrement, est celle de la *segmentation*. Comme nous l'avons mentionné, la segmentation dans le contexte de la vision artificielle est un processus de division d'une image en ses parties.

Formellement, ce processus est défini de la manière suivante : En supposant que l'image complète est représentée par R, et composée de n parties disjointes non vides R_i $(i = 1, 2, \ldots, n)$, les conditions suivantes doivent être verifiées [Fu and Mui, 1981] :

1. $\bigcup_{i=1}^{n} R_i = R$;
2. Pour tout i et j, tel que $i \neq j$,$R_i \cap R_j = \emptyset$;
3. Pour tout $i = 1, 2, \cdots, n$, on a $P(R_i)$ est vrai ;
4. Pour tout $i \neq j$, on a $P(R_i \cup R_j)$ est faux.

Où $P(R_i)$ est un prédicat d'homogénéité pour tous les éléments de l'ensemble R_i.

Les méthodes de segmentation d'images peuvent être classées en trois catégories :

Les méthodes par détection de contours : qui recherchent les frontières entre les régions dans l'image.

Les méthodes par détection de régions : dans lesquelles les pixels voisins sont regroupés selon des critères de similarité.

Les méthodes par classification de pixels : qui considèrent le problème de la segmentation comme un problème de classification de données.

De nombreux travaux de recherche sont réalisés chaque année pour résoudre le problème de segmentation d'images. Cependant, ce dernier n'est toujours pas résolu. Ceci est dû a plusieurs facteurs, notamment, la diversité des spécificités de chaque image, et la définition encore subjective d'une *« bonne » segmentation*, qui dépend des résultats des traitements ultérieurs sur l'image (résultat d'analyse), et surtout du facteur de décision humain.

1.2 Motivations

Les méthodes de segmentation existantes ne sont en général applicables que sur un type d'image particulier et il est difficle de trouver une méthode qui donne de bons résultats pour n'importe quel type d'image. Nous avons donc pensé à cibler un type bien spécifique d'images : les images médicales, qui sont des images en niveaux de gris, constituées de structures anatomiques connues.

Nous considérons la segmentation d'images comme un problème de classification de données, dont le propos est de regrouper les pixels similaires en termes d'intensité dans la même classe. Ce qui produit des ensembles de pixels représentants les différentes structures anatomiques présentes dans l'image médicale.

Le problème de classification de données est un problème NP-difficile. Ainsi, classifier un ensemble de données d'une façon optimale, revient à parcourir l'espace de toutes les solutions possibles, et à calculer la qualité de la classification à chaque étape pour ne garder que la meilleure solution.

Cette solution dite « de force brute » est impraticable dans le cas de la segmentation à cause du temps de calcul trop important même pour de petites images. Le recour à d'autres méthodes qui convergent vers la solution optimale, sans forcement l'atteindre, s'imposent vite dans ce cas. Ces méthodes sont connues sous le nom de *métaheuristiques*.

1.3 Résolution par des algorithmes biomimétiques

Parmi les méthodes visant à résoudre des problèmes NP-difficiles d'une façon générale et les problèmes de classification de données d'une façon particulière, il existe une classe d'algorithmes dits « *biomimétiques* » qui ont la particularité d'être inspirés de comportements observés dans la nature. Un exemple de tels comportements est le déplacement collectif des oiseaux migrateurs ou les réactions des systèmes immunitaires des vertébrés.

Dans de nombreux systèmes naturels composés d'individus à capacités limitées, des tâches complexes peuvent être réalisées sans contrôle global. Ce phénomène est connu sous le nom d'*émergence*, où le comportement global d'un ensemble n'est pas prédictible à partir de l'analyse d'un individu. Dans la nature, ce phénomène d'émergence peut être observé chez des insectes tels les fourmis et les abeilles pour des tâches de collecte de nourritures et de construction des nids, ou chez les oiseux migrateurs et les bancs de poissons pour les déplacements collectifs, ou encore dans notre propre corps tels les réactions immunitaires et les comportements neurologiques.

Dans le présent mémoire, nous allons explorer cette voie, et adapter des méthodes inspirées de comportements naturels pour résoudre le problème de la segmentation. Nous allons de plus, permettre de combiner plusieurs algorithmes de segmentation les uns à la suite des autres de façons à améliorer les résultats.

1.4 Structure et contenu du mémoire

Ce mémoire est devisé en sept chapitres :

Chapitre 1 : Introduction générale Le présent chapitre à pour but de présenter la problématique de ce mémoire, et de montrer le chemin à suivre pour sa résolution.

Chapitre 2 : Introduction au traitement d'images Nous décrirons dans ce chapitre des notions fondamentales du traitement d'image, dans le but de familiariser le lecteur avec le domaine de l'image numérique.

Chapitre 3 : La segmentation d'images Ce chapitre décrit d'une manière détaillée le problème de la segmentation d'images et celui de la classification de données. Nous présenterons, également, les méthodes de segmentation existantes.

Chapitre 4 : Introduction aux méthodes d'optimisation Ce chapitre présente les différentes solutions existantes pour la résolution d'un problème d'optimisation.

Chapitre 5 : Conception et mise en œuvre Dans ce chapitre, la conception de notre application sera détaillée ainsi que les différents algorithmes implémentés.

Chapitre 6 : Tests et Résultats Ce chapitre sera consacré aux tests et aux résultats obtenus, ainsi que leurs interprétations.

Chapitre 7 : Conclusion générale Ce chapitre porte sur une conclusion générale de notre projet. Il énonce également les perspectives de développement qui pourraient lui être apportées.

Chapitre 2

Introduction au traitement d'images

2.1 Introduction

L'« image », au sens le plus général, englobe tout média pouvant être analysé par l'œuil humain. Cela comprend les images fixes, les vidéo, les animations, les graphiques, les diagrammes, les textes manuscrits, etc. L'être humain obtient la plupart des informations sur son environnement à partir des images.

Avec le développement de la technologie ces dernières décennies, le besoin de traiter des images s'est vite avéré indispensable dans plusieurs domaines tels la médecine, le domaine militaires ou la météorologie.

Dans ce qui suit, nous présentons quelques notions fondamentales relatives au traitement et à l'amélioration des images.

2.2 Notions fondamentales

Dans cette section, nous présentons quelques notions fondamentales relatives au domaines du traitement d'images. Pour une présentation plus complète voir [Gonzalez and Woods, 2002].

2.2.1 Image numérique

L'image peut être définie comme étant une fonction à deux variables $f(x, y)$, où x et y représentent les coordonnées spatiales de l'image, et où la valeur $f(x, y)$ représente l'intensité de l'image au point (x, y).

Une autre définition de l'image numérique, est qu'elle peut être considérée comme étant un ensemble fini de données numériques (ensemble de «1» et de «0»), stocké dans un format déterminé.

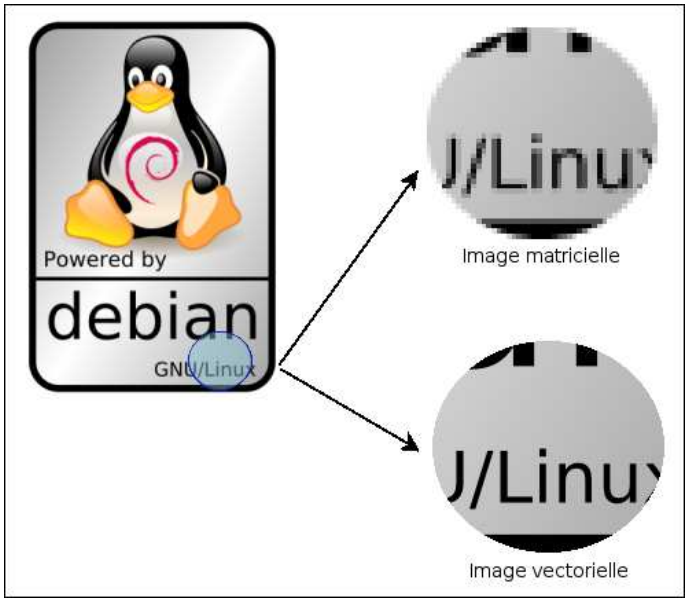

FIG. 2.1 – Effet de l'agrandissement d'une image vectorielle par rapport à une image matricielle

On distingue deux types d'images numériques : les *images matricielles*, et les *images vectorielles*.

2.2.1.1 Image matricielle

Une image matricielle, aussi appelée « image bitmap », est une image numérique représentée par une matrice de pixels.

Une image matricielle est caractérisée par sa résolution, qui définit le niveau de détails visibles dans l'image. Plus le nombre de pixels d'une image est grand, meilleure est sa qualité.

Mathématiquement, une image matricielle est une fonction à deux variables de $\mathbb{R} \times \mathbb{R}$ dans $\mathbb{R}$.

Parmi les formats de stockage des images matricielles, on peut citer : BMP, JPEG, PNG, PGM, XCF, DICOM, etc.

Le principal inconvénient des images matricielles est qu'elles ne peuvent être re-dimensionnées sans perte de qualité (voir la figure 2.1).

Cependant, ce type d'images reste le plus répandu dans l'industrie de l'imagerie, compte tenu du fait qu'il se prête plus facilement au traitement numérique.

2.2.1.2 Image vectorielle

Une image vectorielle est une image numérique stockée dans un format définissant les caractéristiques géométriques de l'image, au lieu d'une matrice de pixels. Ces

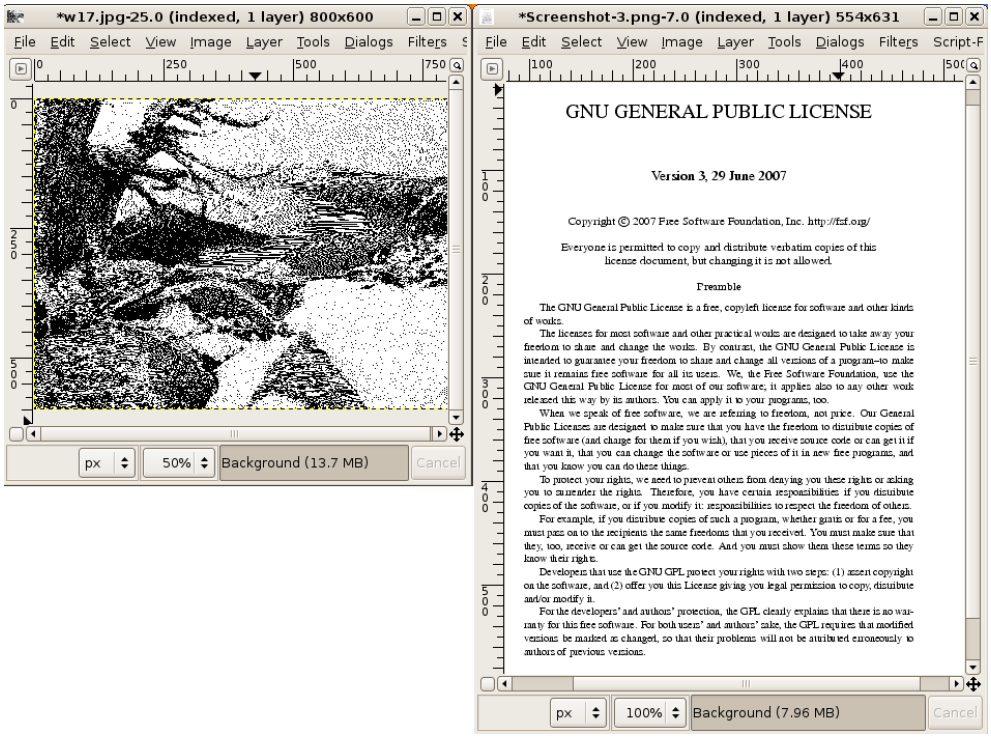

FIG. 2.2 – Deux images binaires

caractéristiques géométriques, peuvent être des lignes, des polygones, des courbes de Bezier, etc.

Il existe différents formats pour stocker des images vectorielles, parmi eux : SVG et WMF.

L'intérêt des images vectorielles, est qu'elle peuvent être re-dimensionnées sans perte de qualité. La figure 2.1 représente l'effet de l'agrandissement d'une image vectorielle par rapport à la même image en format matriciel.

2.2.2 Image binaire

Comme son nom l'indique une image binaire se compose de deux couleurs : le «noir» et le «blanc». Ce type d'image s'applique particulièrement aux images de textes, dans les autres cas la perte en qualité devient trop importante. La figure 2.2 montre deux images binaires : une image texte et une autre capturée par un appareil photographique.

2.2.3 Image en niveaux de gris

Une image en niveaux de gris est une image dont chaque pixel est codé sur 8 bits, d'où un total de 2^8 (256) valeurs possibles pour chaque pixel. Par convention,

FIG. 2.3 – Palette des niveaux de gris.

FIG. 2.4 – Radiographie du crâne en niveaux de gris.

le blanc correspond à la valeur « 255 » et le noir à la valeur « 0 ».

2.2.4 Image couleur

Les images couleurs contiennent une information sur la quantité de lumière rouge vert et bleu émise en chaque point.

2.2.4.1 Images 24 bits ou «couleurs vraies»

Dans ce mode de représentation, chaque pixel est codé sur 24 bits, soit 3 octets représentant les trois composantes d'un pixels : le rouge, le vert, et le bleu. Ce mode est aussi appelé «couleurs vraies».

2.2.4.2 Images indexées

L'allocation de 24 bits par pixel peu devenir contraignante d'un point de vu espace de stockage, surtout pour des images relativement simples ne contenant qu'un nombre restreint de couleurs.

Dans ce cas là, la codification indexée est la plus appropriée. Dans cette dernière, chaque pixel de l'image est codé sur un octet qui représente l'indexe d'une couleur

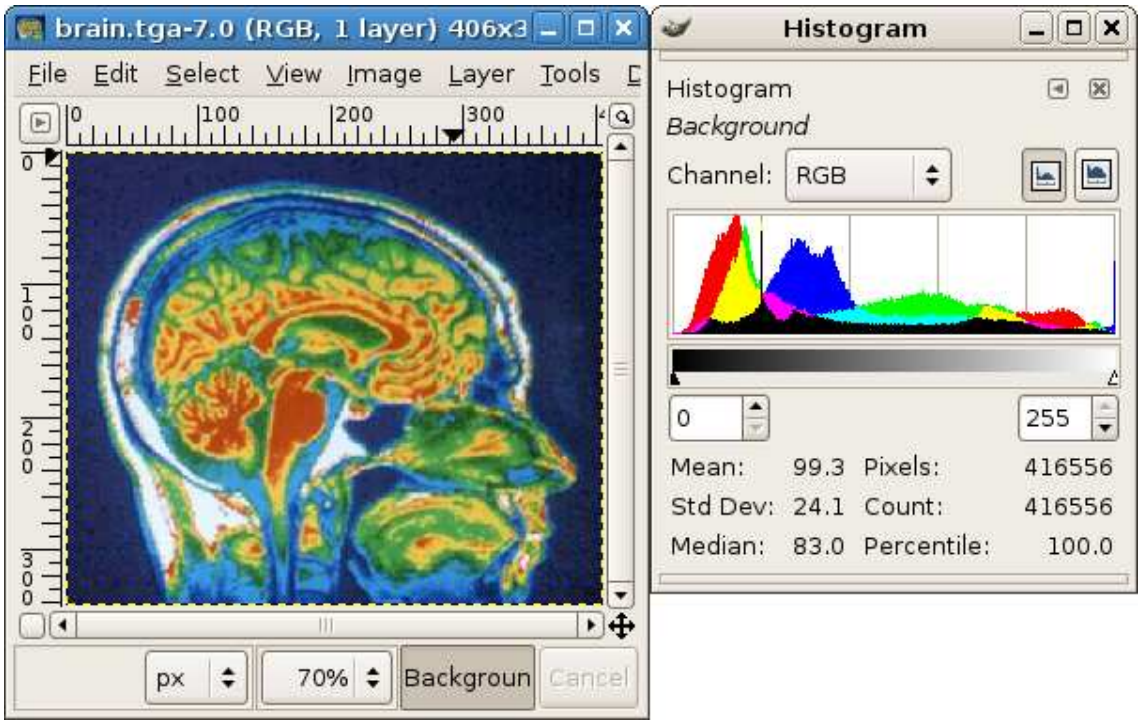

FIG. 2.5 – Image coloriée d'un encéphale, avec la distribution RGB correspondante

dans une table. Le choix de la table des couleurs peut se faire manuellement, ou par programme. Des algorithmes d'optimisation sont utilisés pour déterminer la table optimale des couleurs pour une image donnée.

La figure 2.6 montre la fenêtre d'un logiciel de traitement d'images pour la codification indexée d'une image.

2.2.5 Résolution

La résolution d'une image numérique représente le nombre de pixels par pouce[1] (PPP) (en anglais «Pixel Per Inch (PPI)») en se qui concerne les supports numériques, et par le nombre de Points Par Pouce (PPP)[2], (en anglais «Dots Per Inch (DPI)») en ce qui concerne l'impression papier.

Ainsi, une image de 3×3 pouces, avec une résolution de 300 PPI contient 900×900 pixels.

Le terme «résolution» est aussi employé pour désigner le nombre de pixels composant une image. On parle alors du nombre de pixels par ligne et du nombre de pixels par colonne (voir la figure 2.7) .

[1] 1 pouce=2.54cm

[2] En français, l'abréviation PPP est souvent source de confusion entre Pixels Par Pouce et Points Par Pouce

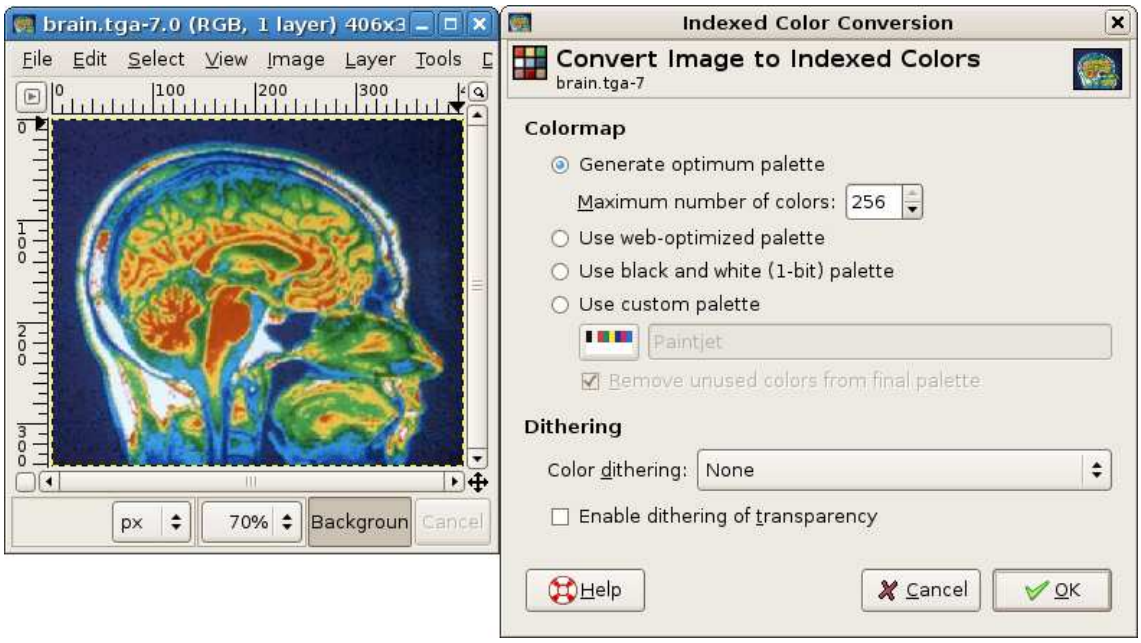

FIG. 2.6 – Choix de la table d'indexation pour la conversion de l'image RGB.

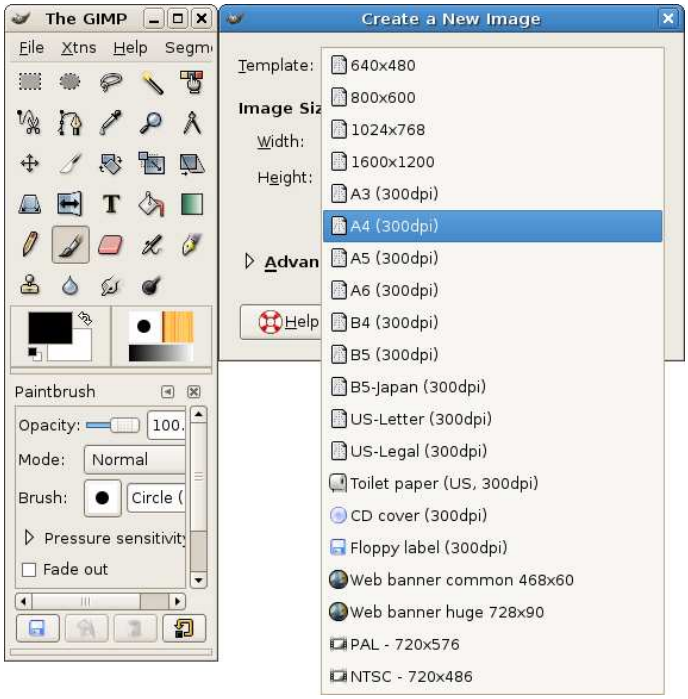

FIG. 2.7 – Sélection de la résolution d'une image.

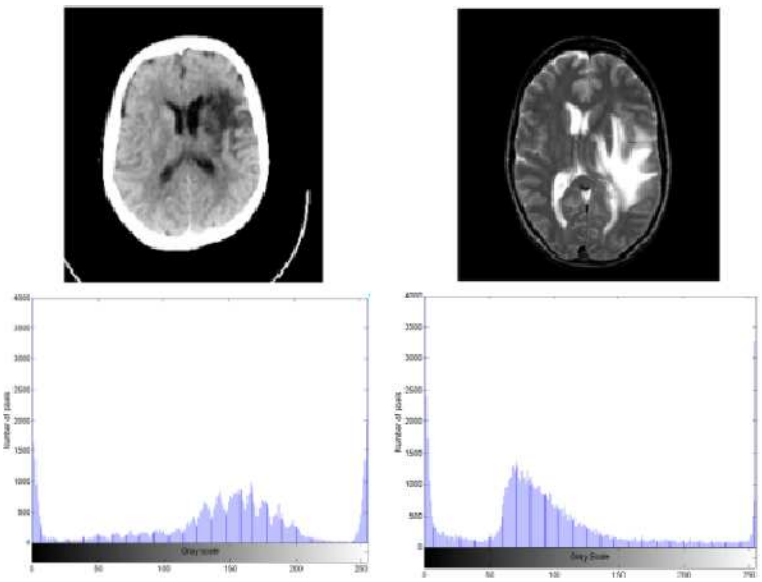

FIG. 2.8 – Une image par rayons X tomographie informatisée (gauche). Image par résonance magnétique (droite) et leur histogramme respectifs[Dhawan, 2003].

2.2.6 Histogramme

L'histogramme d'une image fournit des informations sur la distribution des intensités des pixels dans une image. La forme la plus simple d'un histogramme est un graphe représentant pour chaque niveau de gris le nombre d'occurrences des pixels dans l'image. Mathématiquement, un histogramme $h(r_i)$ peut être défini de la façon suivante :

$$h(r_i) = n_i \text{ pour } i = 0, 1, ..., L-1$$

Où r_i est le i_{eme} niveau de gris dans l'image. Et n_i est le nombre d'occurrences du niveau de gris r_i dans l'image [Dhawan, 2003]. La figure 2.8 montre deux images du cerveau, une par rayons X tomographie informatisée et l'autre par IRM, ainsi que l'histogramme correspondant à chaque image.

Pour les images couleur, un histogramme différent est nécessaire pour la représentation de chaque composante de couleur.

2.2.7 Notion de bruit

Le bruit dans une image numérique est une anomalie apparue lors de l'acquisition de l'image, et donnant lieu à une incohérence dans un voisinage de pixels. La figure 2.9 représente une image bruitée (à gauche) et le résultat après réduction de bruit (à droite).

FIG. 2.9 – Image bruitée (à gauche) et le résultat après réduction du bruit (à droite).

2.2.8 Gestion des couches (layers)

Certains formats de fichiers images peuvent contenir des images numériques composées de plusieurs couches. L'intérêt est de pouvoir gérer plusieurs parties d'une image individuellement, tout en considérant l'ensemble des parties. Un exemple de fichiers prenant en charge la gestion des couches, est le format XCF de GIMP.

La figure 2.10 montre les différentes couches d'une image, et le résultat obtenu après leur fusion.

2.2.9 Gestion de la transparence

La gestion de la transparence dans une image, est définie au niveau de chaque pixel. Ainsi, en plus du codage des couleurs, un octet supplémentaire est ajouté pour gérer la transparence d'un pixel. On parle alors de canal *alpha* ou d'un codage *RGBA* dans le cas du mode RGB avec transparence. Par conséquent, un pixel peut varier de la transparence complète à l'opacité.

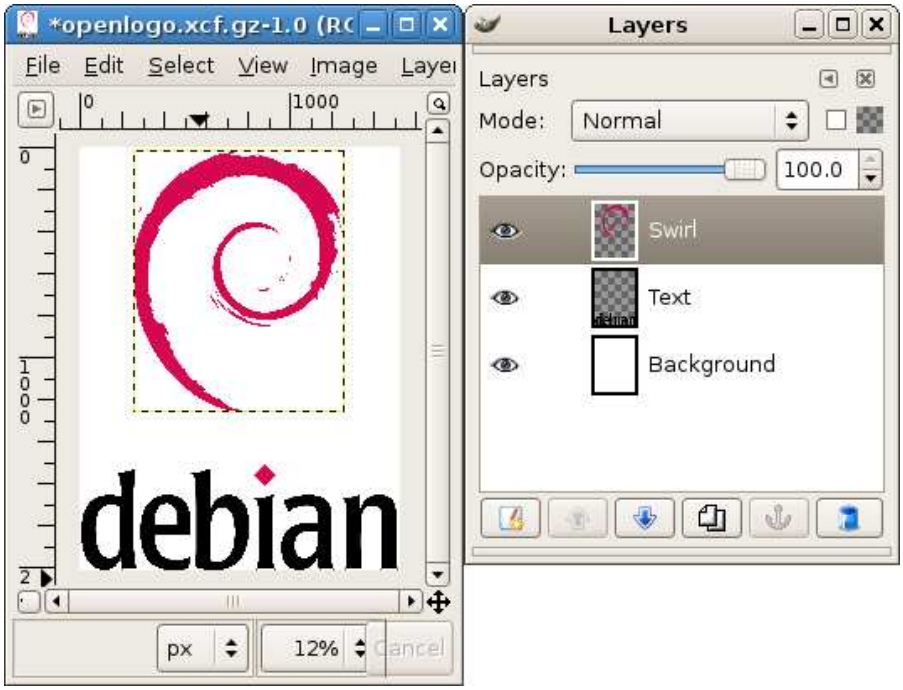

FIG. 2.10 – Image prenant en charge la gestion des couches.

2.3 Opérations de traitement d'images

Dans cette section, nous allons présenter des opérations de traitement d'images dont le but est de les améliorer afin d'augmenter la fiabilité des résultats de l'analyse et de l'interprétation de leurs contenus.

Les opérations à présenter, sont en faite, des méthodes d'amélioration pour corriger quelques erreurs apparues lors de l'acquisition des images. Ces méthodes peuvent être classées en deux catégories [Gonzalez and Woods, 2002] :

1. *Les méthodes basées sur domaine spatial* : Ces méthodes manipulent les valeurs des pixels de l'image dans le domaine spatial.
2. *Les méthodes basées sur le domaine fréquentiel* : Ces méthodes manipulent l'information dans le domaine fréquentiel en se basant sur les caractéristiques fréquentielles de l'image.

2.3.0.1 Méthodes basées sur le domaine spatial

Les méthodes basées sur le domaine spatial traitent l'image pixel par pixel en se basant sur les statistiques de l'histogramme (Voir la définition ci-dessous) ou sur des opérations de voisinage. Ces méthodes sont relativement rapides par rapport aux méthodes fréquentielles qui requièrent le calcul de la transformée de Fourier.

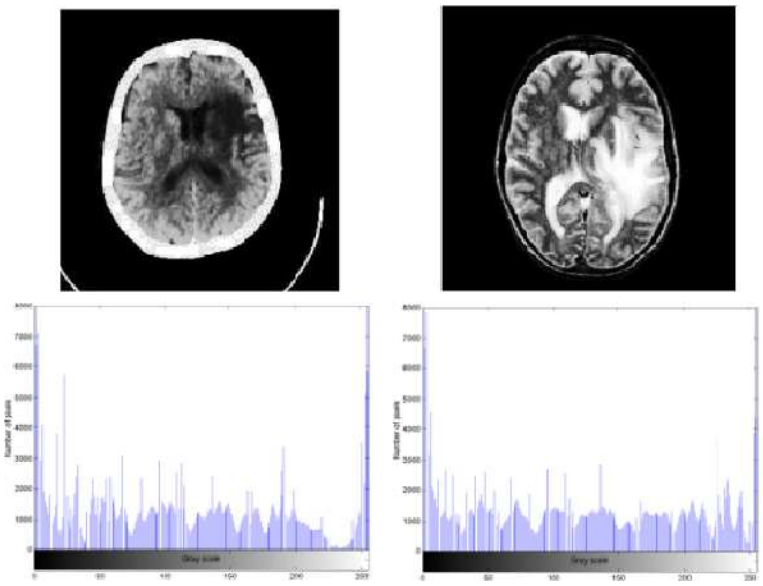

FIG. 2.11 – Histogrammes égalisés de deux images medicales du cerveau [Dhawan, 2003].

Dans cette section, nous allons décrire quelques méthodes d'amélioration basées sur le domaine spacial.

Egalisation de l'histogramme C'est une méthode basée sur la transformation de l'histogramme, afin d'obtenir une image dont l'histogramme représente une distribution uniforme d'intensités des niveaux de gris (voir la figure 2.11).

Moyennage d'images L'acquisition d'une image résulte souvent en une petite dégradation causée par le bruit (anomalie faussant les valeurs des pixels lors de l'acquisition). Calculer la moyenne d'une image réduit le bruit, améliorant ainsi la qualité de l'image. En supposant qu'une image idéale $f(x, y)$ est dégradée par la présence d'un bruit $n(x, y)$, l'image acquise est représentée par :

$$g(x, y) = f(x, y) + n(x, y) \tag{2.1}$$

Généralement, le bruit est aléatoire avec une moyenne nulle. Si une séquence de K images est prise sur le même objet, l'image moyenne est obtenue par :

$$\overline{g}(x, y) = \frac{1}{K} \sum_{i=1}^{K} g_i(x, y) \tag{2.2}$$

Où $g_i(x, y)$, $i = 1, ..., K$ représente la séquence d'images. Si le nombre des images K augmente, la valeur de l'image moyenne se rapproche de $f(x, y)$, réduisant ainsi

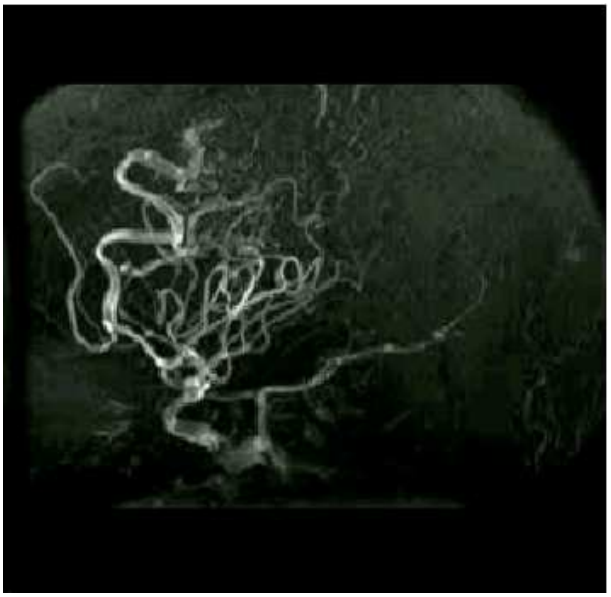

FIG. 2.12 – Angiographie par RM obtenue par soustraction d'images.

–	$f(-1,0)$	–
$f(0,-1)$	$f(0,0)$	$f(0,1)$
–	$f(1,0)$	–

$f(-1,-1)$	$f(-1,0)$	$f(-1,1)$
$f(0,-1)$	$f(0,0)$	$f(0,1)$
$f(0,-1)$	$f(1,0)$	$f(1,1)$

FIG. 2.13 – Voisinage formé d'une connexion de 4 et 8 pixels.

le bruit.

Soustraction d'images Si deux images de la même structure anatomique sont prises dans deux conditions différentes, l'opération de soustraction permet d'améliorer les informations sur les conditions d'acquisition. Cette méthode d'amélioration est appliquée en angiographie, où l'image avec les structures vasculaires est prise dans un premier temps, ensuite, un colorant vasculaire est injecté dans le corps, la deuxième image est prise au sommet de l'effet du colorant. La soustraction des images résulte en une image avec des bons contrastes et une visibilité des structures vasculaires. La figure 2.12 montre la méthode de soustraction.

Opérations de voisinage Les opérations de voisinage consistent en la convolution d'une image avec un masque en vu de l'amelioration de cette dernière. Le niveau de gris de chaque pixel est remplacé par une nouvelle valeur calculée selon le masque appliqué au voisinage du pixel ([Dhawan, 2003]).

1	2	1
2	4	2
1	2	1

FIG. 2.14 – un masque pondéré pour le lissage de l'image utilisant l'équation 2.3.

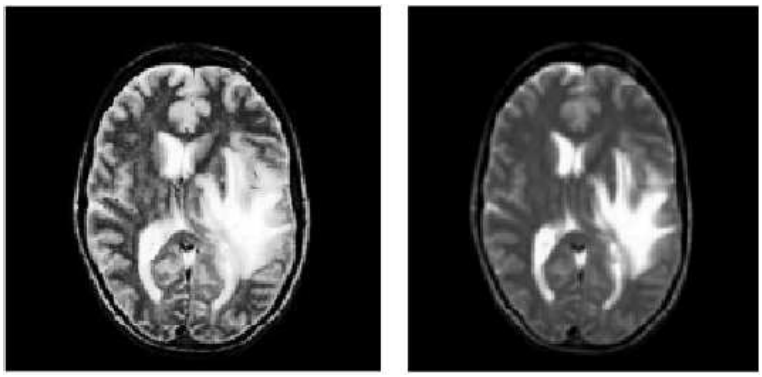

FIG. 2.15 – Image IRM du cerveau (à gauche) lissée (à droite) à l'aide du filtre de la figure 2.14.

Supposons un masque $w(x, y)$ de $(2p+1) \times (2p+1)$ pixels, $p \in N$. La convolution d'une image $f(x, y)$ avec $w(x, y)$ est donnée par :

$$g(x, y) = \frac{1}{\sum_{\acute{x}=-p}^{p} \sum_{\acute{y}=-p}^{p}} w(\acute{x}, \acute{y}) f(x + \acute{x}, y + \acute{y}) \tag{2.3}$$

La convolution est appliquée pour tout x et y de l'image $f(x, y)$. Selon le masque appliqué on peut améliorer différentes caractéristiques de l'image. La figure 2.15 montre le lissage d'une image IRM en appliquant le filtre de la figure 2.14.

Filtre médian Le filtre médian permet l'élimination (ou l'adoucissement) du bruit tout en préservant les contours. Ce filtre remplace la valeur du niveau de gris de chaque pixel de l'image par *la médiane* des niveaux de gris de son voisinage. Ceci se traduit mathématiquement par :

$$\hat{f}(x, y) = median\{g(i, j)\}, \ (i, j) \in N \tag{2.4}$$

Où N est un voisinage prédéfini (par exemple 3×3) du pixel (x, y).

2.3.0.2 Filtrage basé sur le domaine fréquentiel

Les méthodes de filtrage basées sur le domaine fréquentiel manipulent l'image dans le domaine de Fourier (Transformée de Fourier). Ainsi, le traitement se base sur la transformée de Fourier avant de revenir vers le domaine spatial par la transformée de Fourier inverse. On peut distinguer plusieurs types de filtres dans le domaine fréquentiel :

Filtres passe-bas : Ce type de filtres permet la suppression du bruit dans l'image.

Filtres passe-haut : Ce type de filtres permet l'accentuation des détails et du contraste de l'image.

2.4 Conclusion

Dans cette partie du rapport, nous avons décrit quelques notions fondamentales du traitement d'images, dont nous aurons besoin par la suite.

Chapitre 3

La segmentation d'images

Dans ce chapitre nous présentons le problème traité dans ce mémoire : la segmentation d'images. Dans un premier lieu, nous rappelons quelques notions de la vision artificielle. Ensuite nous passons en revu les méthodes de segmentation d'images en nous focalisant sur les méthodes de segmentation par classification, qui sont l'objet de ce mémoire.

Nous terminerons par la présentation de quelques méthodes d'évaluation de la qualité d'une segmentation.

3.1 Introduction à la vision par ordinateur

Le mécanisme de la vision humaine est un processus instinctif mais complexe, dotant l'homme d'une grande capacité de perception et d'interprétation du monde extérieur à partir d'un ensemble d'images. C'est de ce système naturel qu'est venu l'idée de la vision par ordinateur (appelée aussi vision artificielle). Ce système de vision a pour objectif de reproduire artificiellement le système de vision humain en simulant les différentes tâches de perception et d'interprétation du processus naturel.

Les systèmes de vision artificielle sont appliqués dans plusieurs domaines tels que :

- Le processus de contrôle (par exemple dans des robots industriels).
- La modélisation des objets ou des environnements (inspection industrielle, analyse d'images médicales ou modélisation topographique, etc).
- La reconnaissance d'empreintes digitales.
- La détection d'événements (par exemple pour la surveillance visuelle).

Dans un système de vision artificielle, plusieurs étapes sont à considérer. Ces étapes peuvent s'exécuter de façon parallèle ou séquentielle, selon le domaine d'application. D'une manière générale ces étapes concernent :

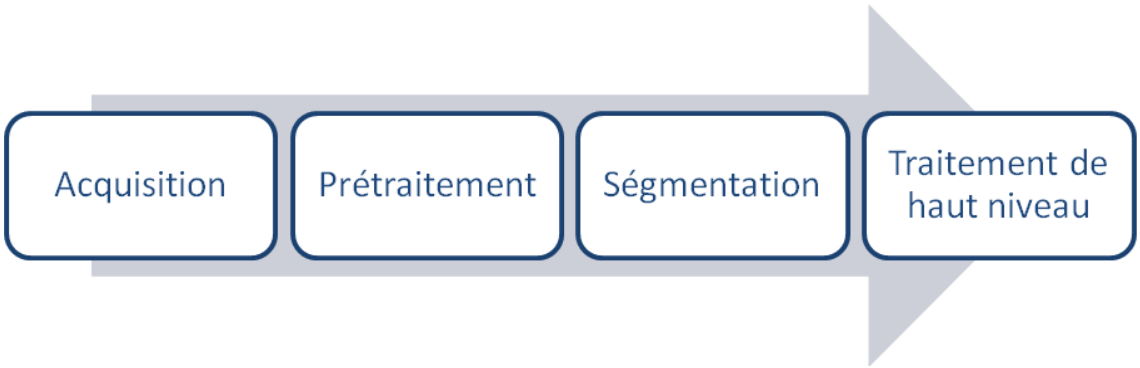

FIG. 3.1 – Schéma d'un système de vision artificiel

L'acquisition de la scène : L'acquisition est la première étape dans un système de vision artificielle. Elle concerne la production d'une image numérique exploitable par ordinateur et joue un rôle très important pour la suite du processus. Les outils le plus souvent utilisés pour l'acquisition sont : les appareils photographiques numériques, les scanners, ou des imageurs spécialisés pour un domaine précis, par exemple les scanners IRM pour l'imagerie médicale.

Le prétraitement : Après l'étape de l'acquisition, l'image numérique a souvent besoin d'être améliorée afin de corriger quelques erreurs apparues lors de l'étape d'acquisition. Cette étape peut donc être considérée comme une étape d'amélioration de la qualité de l'image obtenue. Les méthodes d'amélioration peuvent être classées en deux catégories [Gonzalez and Woods, 2002] :

1. les méthodes basées sur le domaine spatial : Ces méthodes manipulent ponctuellement les valeurs des niveaux de gris des pixels de l'image.
2. les méthodes basées sur le domaine fréquentiel : Ces méthodes se basent les caractéristiques fréquentielles de l'image.

La segmentation : Cette étape permet l'extraction de données pertinentes de l'image en régions ou contours, pour un traitement de plus haut niveau.

Traitement de haut niveau : À cette étape, l'entrée est un ensemble restreint de données, par exemple un ensemble de points ou une région de l'image sensée représenter un objet spécifique. Le traitement faisant suite à cette étape consiste en :

- L'évaluation des paramètres spécifiques à l'application, telle la position ou la taille d'un objet ;
- La classification d'un objet détecté selon différents critères.

Selon la modélisation présentée par la figure 3.1, on peut considérer la segmentation comme une étape de niveau intermédiaire dans un système de vision artificielle.

La segmentation d'images est une étape préliminaire et incontournable. Elle est décrite comme étant le processus de décomposition d'une image en ses différentes parties. C'est une étape critique et primordiale dans tout processus d'analyse automatique d'images, car ses résultats affecteront les étapes suivantes de plus haut niveau comme la reconnaissance et la description des objets. Une définition formelle de la segmentation d'images peut être énoncée comme suit :

En supposant que l'image complète est représentée par R, composée de parties disjointes et non videsR_i $(i = 1, 2, ..., n)$:

1. $\bigcup R_i = R$
2. Pour tout i et j, tel que $i \neq j$,$R_i \bigcap R_j = \emptyset$.
3. Pour tout $i = 1, 2, \cdots, n$, on a $P(R_i)$ est vrai
4. Pour tout $i \neq j$, on $P(R_i \bigcup R_j)$ est faux

Où P est un prédicat d'uniformité.

- la condition (1) indique que l'union des parties doit être l'image originale ;
- La condition (2) indique que les parties sont disjointes ;
- La condition (3) indique que les pixels d'une même région doivent être semblables (selon un critère d'homogénéité donné) ;
- la condition (4) indique que les parties doivent être significativement différentes selon le même critère.

Les premiers développements dans le domaine de la segmentation d'images datent des années 60. En 1965, une méthode de détection des contours visant à séparer les différents objets d'une image fut introduite (sous le nom d'opérateur Robert). Ce fut le premier pas vers la décomposition d'une image en ses composants. Un grand nombre de techniques et de méthodes de segmentation furent introduites depuis. De plus, le concept même de d'« image » a été grandement étendu : Le passage des images statiques vers les séquences d'images (vidéo), des images 2D vers les images 3D, des images en niveau de gris en images en couleur a aussi fortement influencé le développement de la recherche.

Cependant et malgré plusieurs décennies de recherche, la segmentation d'images reste un sujet de recherche d'actualité, et ce pour deux raisons :

- Le nombre de publications et de conférences sur le sujet ne cesse d'augmenter chaque année ;
- La plupart des ouvrages sur la vision par ordinateur, incluent des chapitres sur la segmentation d'images, mais très peu se consacrent entièrement à ce sujet.

Le premier point montre que la recherche dans le domaine est toujours en évolution, et le second, qu'elle n'a pas encore atteint sa maturité.

La recherche sur la segmentation d'images a commencé par le développement de techniques de segmentation, mais en l'absence de fondement théorique standard, beaucoup d'algorithmes sont apparus, et différents principes et conventions ont été adoptés. Il est à noter qu'aucun des algorithmes n'est généralement applicable à tout type d'images, et que différents algorithmes peuvent être plus ou moins adaptés à une application particulière.

3.2 Les méthodes de segmentation

Plusieurs techniques de segmentation d'images existent. Ces techniques peuvent être classées selon plusieurs critères. Elles peuvent être regroupées en trois catégories : l'approche par *détection de contours*, *l'approche par région*, et *l'approche par classification* [Turi, 2001].

3.2.1 L'approche par détection de contours

Dans les techniques basées sur la détection de contours, la segmentation est réalisée par la recherche des frontières délimitant les régions dans l'image. Ceci est généralement réalisé par l'application de masques sur l'image afin de détecter les changements locaux dans l'intensité des pixels. Il existe aussi d'autres méthodes basées sur la détection de contours comme celle des modèles déformables, dans laquelle les frontières d'une région sont délimitées suivant des formes géométriques, préalablement initialisées. Dans ce qui suit nous allons présenter quelques approches de la détection de contours.

3.2.1.1 Méthodes dérivatives

Ceux sont les méthodes les plus utilisées pour la détection des transitions des éléments d'une image. Le principe général est de balayer une image avec un masque définissant la zone d'intérêt [Ouadfel, 2006].

Le résultat obtenu sera alors une image binaire constituée de deux classes : les pixels des contours et les pixels des non-contours.

Ces méthodes sont très sensibles au bruit, et pour obtenir un bon résultat il est recommandé de lisser l'image avant l'application du masque.

On peut classer les méthodes dérivatives selon deux approches :

- Approche Gradient.
- Approche Laplacien.

Détection de contours par l'approche gradient : Les contours dans une image étant caractérisés par une forte variation de contraste, il est logique de chercher un opérateur permettant de caractériser et de repérer les zones où les niveaux de gris augmentent ou diminuent très vite. La dérivée (le gradient) répond tout à fait à ce problème.

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

Soit $f(x, y)$ une fonction continue qui représente l'intensité de chaque point de l'image $p(x, y)$. Le gradient de l'image en ce point est le vecteur à deux dimensions $\bigtriangledown f(x, y)$ défini par ses coordonnées dérivatives verticales et horizontales comme suit :

$$\bigtriangledown f(x, y) = \begin{pmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{pmatrix} \tag{3.1}$$

Pour calculer ce gradient en chaque point de l'image, on effectue, généralement, le produit de convolution de l'image avec un opérateur de dérivation fournissant deux masques $M1$ et $M2$ tels que :

$$\frac{\partial f(x, y)}{\partial x} = G_x(x, y) = M1 \times f(x, y); \tag{3.2}$$

et

$$\frac{\partial f(x, y)}{\partial y} = G_y(x, y) = M2 \times f(x, y) \tag{3.3}$$

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

$$G(x, y) = \sqrt{G_x^2(x, y) + G_y^2(x, y)} \tag{3.4}$$

Et la direction du gradient est donnée par :

$$d(x, y) = \arctan \frac{G_y(x, y)}{G_x(x, y)} \tag{3.5}$$

Il existe plusieurs opérateurs permettant l'approximation du gradient, on peut citer :

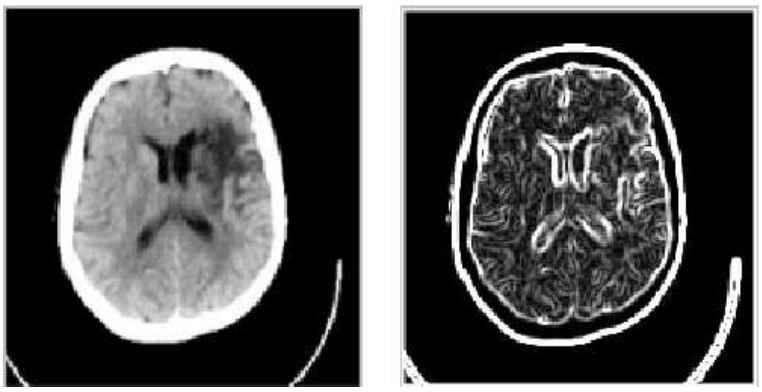

FIG. 3.2 – A gauche, une image du cerveau par par résonance magnétique. A droite, L'application du masque de Sobel pour la détection de contours sur l'image de gauche.

– Masque de **Prewitt** :

$$M1 = \frac{1}{6}\begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix} et\ M2 = \frac{1}{6}\begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & 1 \end{pmatrix} \tag{3.6}$$

– Masque de **Sobel** :

$$M1 = \frac{1}{6}\begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix} et\ M2 = \frac{1}{6}\begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix} \tag{3.7}$$

– Masque de **Roberts** :

$$M1 = \frac{1}{6}\begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} et\ M2 = \frac{1}{6}\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{pmatrix} \tag{3.8}$$

Il existe aussi d'autres approches pour le calcul du gradient non pas dans deux directions uniquement, mais dans quatre directions. Cette catégorie comporte aussi plusieurs types de masques (Sobel, Perwitt,...).

La figure 3.2, montre une image par résonance magnétique et la détection de contours en utilisant le masque de Sobel.

Détection de contours par l'approche Laplacien : L'opérateur gradient du second ordre (Laplacien) peut être calculé par convolution d'un des deux masques : $G_{L(4)}$ et $G_{L(8)}$, qui utilisent, respectivement, 4 et 8 voisinages connectés ;

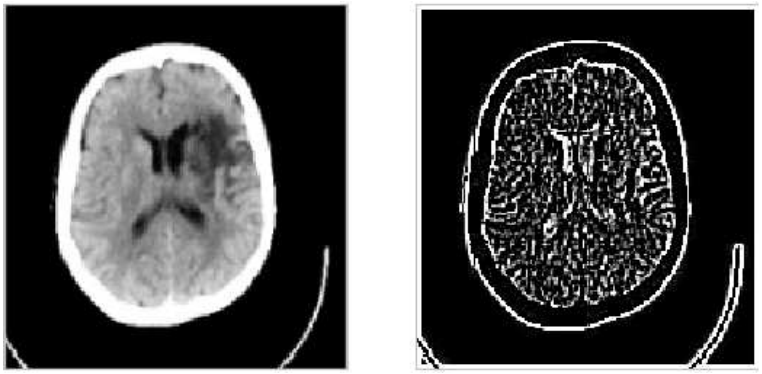

FIG. 3.3 – A gauche, une image du cerveau par résonance magnétique. A droite, L'application du Laplacien pour la détection de contours sur l'image de gauche

$$G_{L(4)} = \begin{pmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{pmatrix} \tag{3.9}$$

$$G_{L(8)} = \begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix} \tag{3.10}$$

Cette approche permet une approximation des passages par zéro de la dérivée seconde de l'image, qui détermine les transitions des intensités des pixels. La figure 3.3 représente l'application du masque Laplacien sur une image en niveau de gris du cerveau.

L'avantage des méthodes dérivatives pour la détection de contours, est leur facilité d'implémentation, leur temps de calcul relativement court, et leur résultats satifaisants pour des images non bruitées.

Leur inconvénient est qu'elles sont très sensibles au bruit. Une manière de résoudre ce problème, est l'utilisation d'un filtre gaussien avant d'appliquer le filtre Laplacien, comme celui utilisé par Marr et Hildreth [Marr and Hildreth, 1980].

Pour pallier au problème des images bruitées, plusieurs opérateurs de dérivation dits « optimaux » sont apparus. Ces opérateurs permettent de détecter les contours tout en respectant les trois conditions suivantes [Canny, 1986] :

- Bonne détection des contours.
- Bonne localisation des contours.
- Faible sensibilité au bruit.

Parmi ces opérateurs «optimaux», on trouve l'opérateur de Canny [Canny, 1986], et celui de Deriche [Deriche, 1990].

L'opération de détection des contours présents dans l'image est suivie de l'application de procédures pour lier ces contours afin d'assembler les frontières, et former des régions closes.

Les procédures de liaison des contours sont basées sur une recherche effectuée pixel par pixel pour trouver la connectivité parmi les segments des contours.

De plus, des propriétés topographiques sont utilisées pour améliorer les opérations de liaison des contours pour les pixels affectés par le bruit. Des méthodes d'estimation basées sur des approches probabilistes ainsi que des méthodes basées sur des graphes ont aussi été utilisées.

3.2.1.2 Méthodes utilisant les techniques incorporant des propriétés globales

Dans des situations plus complexes, les techniques de bas niveau ne sont pas efficaces et ce principalement à cause de la non-homogénéité des objets à segmenter et du fait que ces derniers peuvent contenir des bords et des coins pouvant fausser les résultats d'un détecteur de contours. Dans des situations telles que celle-ci l'incorporation de connaissances sur la structure globale des objets dans l'algorithme devient nécessaire.

Modèles déformables : La théorie des modèles déformables (ou contours dynamiques) vise à faciliter l'incorporation de connaissances globales dans un algorithme de segmentation. Ce modèle est largement employé dans le domaine de l'imagerie médicale et l'étendue de ses possibilités n'est pas encore complètement explorée.

Un modèle déformable peut être considéré comme un contour fermé, définit par des points de contrôle qui s'étend petit à petit jusqu'à recouvrir tout l'objet. Ce qui rend les modèles déformables intéressants est le fait qu'ils ne se basent pas ou peu sur les propriétés locales des images pour contrôler leur extension, mais sur des contraintes à priori imposées sur les mouvements des points de contrôle et sur les interactions entre ces derniers. Ces contraintes reflètent les propriétés globales de l'objet en question. La figure 3.4 montre un exemple d'application des modèles déformables.

Trois facteurs principaux interviennent pour contrôler l'extension d'un modèle déformable.

- Les points de contrôle interagissent les uns sur les autres pour maintenir une forme lisse qui ne comporte pas de cassures trop importantes ;
- Le contour doit s'adapter le plus possible aux bords, coins et différentes propriétés de l'image.

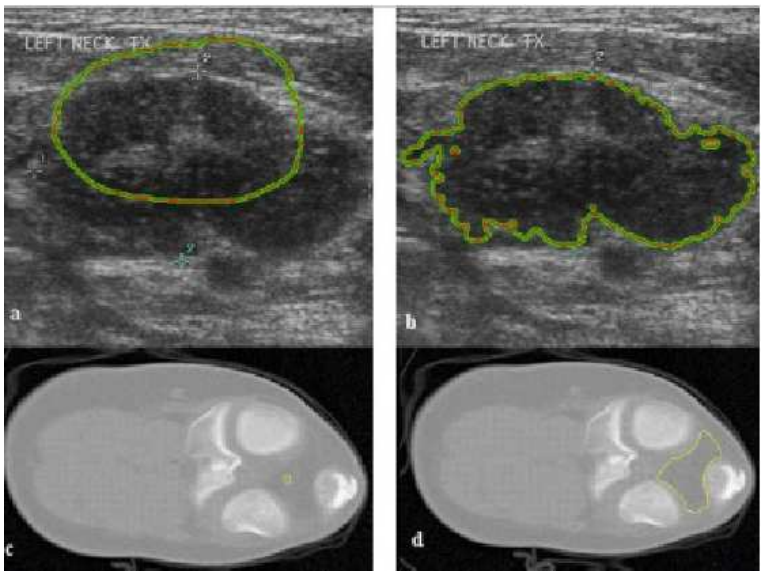

FIG. 3.4 – Application des modelés déformables aux données médicales. La colonne de gauche montre l'état initial du contour, celle de droite le résultat. (a) et (c) représentent une tumeur du larynx. (b) et (d) un scanner CT du genou.

- Les propriétés globales de l'objet à segmenter (qui conditionnent tout le processus).

Ces trois facteurs sont incorporés dans des équations qui définissent le mouvement des points de contrôle.

3.2.2 Segmentation par régions

Les algorithmes basés sur l'extension des régions examinent les pixels voisins pour rechercher d'éventuelles similarités suivant un critère donné. Les pixels voisins répondant au critère de similarité sont regroupés pour former les régions. Les techniques d'extension de régions peuvent être généralisées au regroupement de régions et non seulement à celui de pixels.

Une autre approche consiste à démarrer avec de « grandes » régions puis à les partitionner jusqu'à obtenir des régions homogènes.

La différence principale entre les méthodes basées sur les régions et les méthodes de classification de pixels est que les premières ne regroupent que les pixels voisins, alors que les secondes peuvent produire des régions non connexes.

En plus de la ressemblance en termes des intensités des pixels d'une région, le critère d'homogénéité intra-classes est étendu à l'information spatiale qui joue un rôle important dans le processus de segmentation. D'où la définition d'un prédicat d'homogénéité des régions, intègrant, en plus d'un critère de ressemblance de niveaux

de gris, un critère de ressemblance spatial, ce qui rend la définition d'un critère d'homogénéité relativement difficile par rapport à l'approche par classification qui ne prend pas en compte le critère spatial.

L'approche de segmentation par région comprend trois catégories de méthodes :

- Les méthodes par fusion de régions ;
- Les méthodes par division de régions ;
- Les méthodes par fusion/division de régions.

Dans ce qui suit nous décrivons brièvement chacune des trois approches.

3.2.2.1 Méthodes par fusion de régions

La fusion de régions est une approche reposant sur le regroupement itératif d'un ensemble de régions uniformes d'une image. Partant d'un ensemble de régions de dimension réduite, la fusion de régions adjacentes se fait au fur et à mesure, selon un critère d'homogénéité prédéfini. Dans cette catégorie d'algorithmes, la fusion des régions d'une image est généralement réalisée à l'aide d'un graphe d'adjacence, dans lequel les nœuds représentent les régions, et les arêtes l'adjacence des régions : une arête entre deux nœuds signifie que les deux régions représentées par les deux nœuds sont adjacentes dans l'image.

3.2.2.2 Méthodes par division de régions

Dans cette approche, l'image originale est divisée récursivement en régions jusqu'à ce que le critère d'homogénéité des régions soit vérifié. Au fur et à mesure des étapes de division des régions, chaque région est évaluée selon le critère d'homogénéité. Si la région est homogène la division s'arrête pour cette région, sinon on réitère l'opération jusqu'à ce que le critère soit vérifié. La division d'une image est réalisée principalement selon deux modèles géométriques :

Arbre quartenaire : C'est une structure d'arbre dont chaque nœud possède quatre nœuds fils. Dans cette structure, l'image principale est représentée par la racine, les différentes portions de l'image (jusqu'aux pixels) sont représentées par les nœuds de l'arbre, jusqu'aux feuilles qui représentent des portions homogènes. La division de l'image consiste à tester son homogénéité, et dans le cas ou ce critère n'est pas satisfait, à la découper en portions selon la structure de l'arbre quartenaire. L'opération de division est réitérée jusqu'à ce que le critère d'homogénéité soit vérifié sur chaque portion de l'image.

L'inconvénient de cette méthode de division est sa rigidité [Ouadfel, 2006], et sa forte concentration sur des régions de forme carrée.

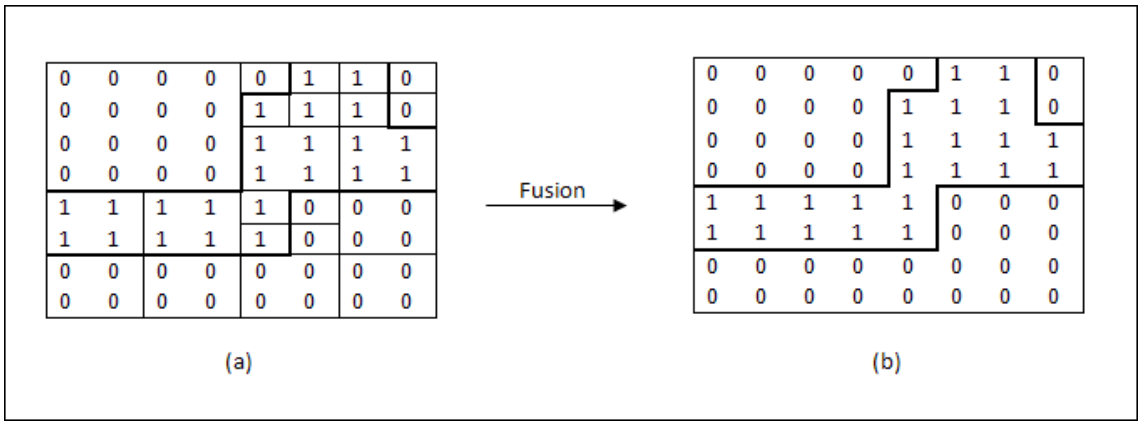

FIG. 3.5 – (a) Images binaire après division par arbre quartenaire, (b) fusion des régions similaires de l'image (a).

Structure de Voronoï : Comme dans le modèle précédent, l'image est divisée récursivement mais non pas suivant une structure d'arbre, mais selon un découpage de polygones dans l'image jusqu'à ce que le critère d'homogénéité soit satisfait par toutes les portions de l'image.

3.2.2.3 Méthodes par division/fusion

Dans ces méthodes, la segmentation est basée sur la combinaison des deux modèles précédents. Dans une première étape l'image est divisée de telle façon à obtenir des portions homogènes. Le résultat de cette étape sert de point de départ à la méthode de fusion, qui rassemble les régions semblables selon un critère donné. Dans la figure 3.5, une image binaire est divisée selon la méthode de l'arbre quartenaire, qui divise l'image récursivement par blocs de quatre régions jusqu'à la stabilité de toutes les régions. Une région est dite stable si elle est homogène, soit, dans le cas d'une image binaire si tous ses pixels sont semblables.

Les régions obtenues par division sont fusionnées selon leur ressemblance, pour obtenir en finalité deux régions homogènes.

3.2.3 Segmentation par classification

La segmentation d'images par classification est une approche inspirée du problème de partitionnement des données (data clustering). L'image est divisée en un ensemble de classes dont chacune rassemble des pixels ayant des caractéristiques communes.

La classification des pixels ne se base pas sur leurs positions spatiales, contrairement aux méthodes par régions, mais sur leurs caractéristiques statistiques. Après

une segmentation, chaque pixel de l'image résultante reçoit une étiquette de la classe à laquelle il appartient. Par conséquent, une même classe peut être constituée de régions se trouvant à des endroits différents dans l'image.

Dans ce qui suit nous allons présenter de façon détaillée le problème de partitionnement des données, qui est une approche générale de la segmentation par classification pour laquelle les données ne sont autres que les pixels de l'image.

3.2.3.1 Problème de partitionnement des données

Le partitionnement des données peut être défini comme un processus de classification de données multidimensionnelles en différents groupes. Les données appartenant au même groupe présentent des caractéristiques communes se traduisant souvent par un critère de proximité évalué selon des mesures de distance.

Le partitionnement des données est un thème central dans le domaine de l'intelligence artificielle [Hamerly, 2003], il est présent dans beaucoup d'applications telles que

- La reconnaissance de formes ;
- La quantification d'images couleurs ;
- L'exploration des données (data mining) ;
- Ou encore la segmentation d'images.

Une classe est définie par son centre (centroid), ce qui permet la mise en œuvre d'algorithmes se basant sur la recherche des meilleurs centres pour les données à classifier.

3.2.4 Mesures de similarité

Dans le processus de partitionnement des données, les données appartenant à un groupe présentent une certaine ressemblance qui se formalise par une « petite distance » entre les données d'une même classe. Cette distance est basée sur la « différence » entre les caractéristiques des données. La distance la plus utilisée dans les approches de classification est la distance Euclidienne, qui se définie par :

$$d(x_v, x_w) = \sqrt{\sum_{i=1}^{N}(x_{v,i} - x_{w,i})^2} = \|x_{v,i} - x_{w,i}\| \tag{3.11}$$

Oú N est la dimension du vecteur des caractéristiques d'une donnée. Il existe par ailleurs d'autres mesures de distance telle que la distance de Manhattan, définie par :

$$d(x_v, x_w) = \sum_{i=1}^{N} |x_{v,i} - x_{w,i}| \tag{3.12}$$

3.2.5 Évaluation de la classification

L'objectif principal d'une évaluation de classification est d'en estimer la qualité afin de déterminer le meilleur partitionnement pour un ensemble de données.

L'évaluation consiste donc à affecter un indice d'estimation à un partitionnement donné. Certains algorithmes de classification évaluent l'indice d'estimation au fur et à mesure de leur progression. La solution optimale du partitionnement sera la solution dont l'indice d'estimation est le meilleur.

Dans la littérature, on trouve différents indices d'estimations qui considèrent, pour la plupart, deux critères d'évaluation [Halkidi et al., 2001] :

Compacité : Les données appartenant à une classe doivent répondre à un critère de similarité. Par exemple, la variance des données d'une classe donne une indication sur la compacité.

Séparation : Les classes doivent être bien séparées, la distance euclidienne entre les centres des classes donne une indication sur la séparation.

Dans ce qui suit nous allons présenter trois indices d'estimation d'une classification. Ces indices seront utilisés dans la suite de notre travail pour l'évaluation de la qualité des résultats des algorithmes.

3.2.5.1 Indice de Dunn

Le but de l'utilisation de cet indice est de maximiser la distance interclasses (séparation), et de minimiser la distance intra-classes (compacité). Il est défini comme suit :

$$D = \min_{k=1,\ldots,K} \min_{kk=k+1,\ldots,K} \left(\frac{dist(C_k, C_{kk})}{\min_{k=1,\ldots,K} diam(C_a)} \right) \tag{3.13}$$

Oú $dist(C_k, C_{KK})$ représente l'indice de dissimilitude entre deux classes qui est défini comme suit :

$$dist(C_k, C_{KK}) = \min_{u \in C_k, w \in C_{kk}} d(u, w) \tag{3.14}$$

Oú $d(u, w)$ est la distance euclidienne entre u et$w$. L'expression $diam(C)$ correspond elle au diamètre de la classe C, définie par :

$$diam(C) = \max_{u \in C, w \in C} d(u, w) \tag{3.15}$$

Une classification optimale est celle qui maximise l'indice de Dunn.

3.2.5.2 Indice de Davis et Boldin

Cet indice permet de minimiser la similarité moyenne entre une classe et celle qui lui est le plus proche ([Davis and Bouldin, 1979]). Cet indice est défini comme suit :

$$DB = \frac{1}{K}\sum_{i=1}^{K} \max_{j=1,\ldots,K/i\neq j} \left(\frac{diam(C_i) + diam(C_j)}{dist(C_i, C_j)} \right) \tag{3.16}$$

Une classification optimale est celle qui minimise l'indice DB.

3.2.5.3 Indice de Turi

Un autre indice proposé par Turi en 2001 [Turi, 2001] dans lequel apparaît un paramètre aléatoire. Le but de ce paramètre est de ne pas se focaliser exclusivement sur les solutions localement optimales qui peuvent conduire à une convergence prématurée dans certains algorithmes de classification. La formule de l'indice de Turi est la suivante :

$$V = (c \times N(2,1) + 1) \times \frac{intra}{inter} \tag{3.17}$$

Où C est un paramètre défini par l'utilisateur, $N(2,1)$ est une distribution Gaussienne de moyenne égale à 2 et de variance égale à 1. Le terme *intra* correspond à la moyenne entre chaque donnée et son centre dans la classe, formellement :

$$intra = \frac{1}{N_p}\sum_{i=1}^{K}\sum_{\forall u \in C_i} \|u - m_i\|^2 \tag{3.18}$$

Le terme *inter* correspond à la distance entre les deux centres les plus proches, ce qui se traduit par :

$$inter = \min \|m_i - m_j\|^2, \forall i = 1, \ldots, K-1 \; et \; \forall j = i+1, \ldots, K \tag{3.19}$$

3.2.6 Techniques de partitionnement des données

Les techniques de partitionnement des données peuvent être regroupées en deux grandes familles d'algorithmes :

- Les algorithmes de classification hiérarchique ;
- Les algorithmes de classification par partition.

Dans ce qui suit, nous détaillons chacune de ces techniques et illustrons par des exemples d'algorithmes chacune d'elles.

3.2.6.1 Techniques de Classification hiérarchique

Dans les techniques de classification hiérarchique, le partitionnement est réalisé à l'aide d'une matrice de similarité qui contient toutes les distances entre les données. Le résultat obtenu est un dendrogramme qui représente les différents groupements des données, ainsi que les niveaux dans lesquels les groupements ont été modifiés.

Les techniques de classification hiérarchique peuvent être divisées en deux classes :

Les méthodes agglomératives : qui commencent avec un nombre de classes initial égal au nombre de données à classifier. Puis, elles recherchent les deux classes les plus similaires (à l'aide de la matrice de similarité) pour les fusionner. La matrice de similarité est ensuite mise à jour.

Les méthodes hiérarchiques divisives : Il n'existe au départ qu'une seule classe qui contient l'ensemble des données à classifier. Par la suite, la classe est divisée récursivement sur plusieurs étapes selon les distances entre les données. Les méthodes hiérarchiques divisives peuvent être considérées comme le processus inverse des méthodes agglomératives.

Les étapes de base d'une classification hiérarchique agglomérative, peuvent être résumées comme suit :

1. Construire la matrice de similarité contenant les distances entre chaque paire de données.
2. Assigner chaque donnée à une nouvelle classe ($K = N$).
3. Trouver la paire de classes la plus proche C_i, C_j.
4. Fusionner C_i et C_j, supprimer C_j, et $K = K + 1$.
5. Si $K > 1$, aller à 3, sinon Fin.

La distinction entre deux méthodes de classification hiérarchique porte sur la façon de déterminer la similarité entre deux classes. Nous citons quelques méthodes pour illustrer cette notion :

La méthode du lien unique (the single link)[Sneath and Sokal, 1973] : qui considère que deux classes sont similaires si la distance entre leurs plus petits éléments est minimale.

La méthode du lien complet (the complete link)[Anderberg, 1973] : qui, à l'opposé de la précédente, regroupe deux classes si la distance entre leurs deux éléments les plus éloignés est minimale.

La méthode du centroid : qui considère que deux classes sont similaires si la distance entre leurs centroids est minimale.

Les méthodes de classification hiérarchique ont deux avantages principaux (voir [Frigui and Krishnapuram, 1999]) :

1. Le nombre de classes n'est pas spécifié au préalable.
2. Il n'y a pas de problème d'initialisation des partitions.

Elles souffrent par contre des inconvénients suivants [Amrane, 2004] :

1. Coût en calcul (complexité de l'ordre de $O(N^2 logN)$) ;
2. Rigidité, les éléments affectés ne peuvent pas changer de classe ;
3. Manque d'informations sur la forme ou la taille des classes.

3.2.6.2 Techniques de Classification par partitionnement

L'un des inconvénients des méthodes hiérarchiques, est la rigidité dans l'affectation des données : si un élément est affecté par erreur, il n'y a aucun moyen pour le réassigner à la bonne classe. Ce qui peut devenir très contraignant si plusieurs éléments sont affectés par erreur.

Contrairement aux techniques de classification hiérarchiques, les techniques de classification par partitionnement n'ont pas ce problème d'immobilité des données. Les éléments peuvent être réaffectés à différentes classes si cela améliore le partitionnement. Les techniques de classification par partitionnement sont des méthodes itératives qui convergent vers des solutions localement optimales par l'optimisation d'une fonction de fitness qui reflète la qualité de la classification.

Le problème avec ces techniques est qu'elles ont besoin de connaitre le nombre de classes à priori, et que le résultat de la classification dépend fortement de la solution de départ.

Dans ce qui suit nous passons en revue les méthodes de partitionnement les plus connues.

Algorithme K-means : L'un des plus simples et des plus répandu des algorithmes de partitionnement est le K-means [Forgey, 1965]. Partant d'une partition initiale qui représente les centroids des classes, l'algorithme réaffecte les données itérativement en optimisant une fonction objective qui correspond à l'inertie intra classe, exprimée sous la forme :

$$I_{intra} = \sum_{k=1}^{K} \sum_{\forall z_p \in C_k} d^2(z_p, m_k) \tag{3.20}$$

L'algorithme 1 présente le schéma général du K-means tel que décrit par Tou et Gonzalez [Tou and Gonzalez, 1974]. La condition d'arrêt est satisfaite lors-

Algorithme 1 Algorithme K-means.

Initialiser K centres à des valeurs arbitraires.
tantque le critère d'arrêt n'est pas vérifié **faire**
  Affecter chaque élément z_p à une classe, à l'aide de la relation suivante :
  $z_p \in C_k sid^2(z_p, m_k) < d^2(z_p, m_j) j = 1, \ldots, K \ j \neq k$
  Mettre à jour les centres des classes :$m_k = \frac{1}{|c_k|} \sum_{i \in C_k} z_i^k \ k = 1, \ldots, K$
fin tantque

qu'aucun changement n'est observé d'une itération à une autre, ou bien lorsque un nombre maximal d'itérations est atteint.

L'algorithme du K-means et ses variantes exige une connaissance à priori du nombre de classes. Le nombre de classes est dans la plupart des cas choisi arbitrairement, ce qui peut réduire la qualité de la classification. Pour pallier à ce problème, nombreux algorithmes de classification dynamique sont apparus. On peut citer l'algorithme ISODATA [Ball and Hall, 1967] et l'algorithme DYNOC (Dynamic Optimal Cluster seek) [Tou, 1979] qui classifient les données tout en modifiant le nombre de classes au cours des itérations. Le nombre de classes peut augmenter ou diminuer, selon les cas de figures : si la fonction de fitness pour une classe dépasse un seuil fixé par l'utilisateur, la classe est divisée. Si la distance entre deux centroids descend en dessous d'un seuil les deux classes sont fusionnées.

Algorithme Fuzzy C-means Une version floue de l'algorithme du K-means, nommée fuzzy C-means (FCM) a été proposée [Bezdeck, 1981]. Cette approche, comme son nom l'indique, est une méthode de classification basée sur la théorie des ensembles flous, qui décrit un modèle de répartition des données sur des ensembles. L'avantage du FCM sur le K-means, est que chaque élément est affecté à toutes les classes avec des degrés d'appartenance u. Chacun des N éléments à classifier, appartient à chacune des K classes avec un degré d'appartenance. Les degrés d'appartenance aux classes peuvent être décrits dans une matrice d'appartenance U :

$$U = [u_{i,k}]_{i=1,\ldots,N;\ k=1,\ldots,K} \tag{3.21}$$

$u_{i,k}$ étant le degré d'appartenance de l'élément i à la classe k, défini par :

$$u_{i,k} = \left(\sum_{j=1}^{K} \left(\frac{\|z_i - m_k\|}{\|z_i - m_j\|} \right)^{\frac{2}{q-1}} \right)^{-1} \tag{3.22}$$

Le paramètre q correspond au coefficient de flou, plus q est grand plus la partition est floue [Ouadfel, 2006]. Le choix de q reste donc très important pour l'obtention d'un bon résultat de classification.

Les degrés d'appartenance des éléments aux classes $u_{i,k}$ doivent satisfaire aux contraintes suivantes :

1. $\sum_{k=1}^{K} u_{i,k} = 1; \forall i \in [1; N]$;
2. $0 \leq u_{i,k} \leq 1 \forall i \in [1; N], \forall k \in [1; N]$.

La première contrainte signifie qu'un élément appartient à l'union des classes. La deuxième signifie que chaque élément possède un degré d'appartenance à chaque ensemble.

La méthode FCM a pour but de minimiser une fonction de fitness, reflétant la distribution des éléments sur les classes, et définie par :

$$J_{FCM} = \sum_{k=1}^{K} \sum_{i=1}^{N} u_{i,k}^{q} \|x_i - m_k\|^2 \tag{3.23}$$

L'algorithme FCM donne généralement de meilleurs résultats que le K-means [Hamerly, 2003]. Cependant, comme pour le K-means, le nombre de classes doit être spécifié préalablement, et la convergence se produit souvent vers un optimum local.

3.3 Méthodes d'évaluation d'une segmentation

Beaucoup de méthodes d'évaluation de segmentation ont été proposées ces dernières décennies.

Un problème majeur dans la segmentation, est la diversité des types de régions composant une image. En effet, une image peut se composer de régions uniformes, texturisées ou dégradées. Peu de méthodes de segmentation fournissent de bons résultats pour chaque type de région. En effet, l'efficacité d'une nouvelle méthode de segmentation est souvent illustrée par seulement quelques résultats sur des images de référence, telles que l'image Lena.

Le problème est que cette évaluation basée sur des benchmarks spécifiques est encore subjective. Ainsi, la comparaison des différentes méthodes de segmentation

n'est pas une tâche facile. De ce fait et pour parvenir à une comparaison objective des différentes méthodes ou des résultats de segmentation, quelques critères d'évaluation ont été définis suivants deux approches :

L'évaluation basée sur des critères supervisés : Ces critères se basent sur le calcul de mesures globales de dissimilitude entre une segmentation de référence et les résultats obtenus après le processus de segmentation. Dans cette approche, deux composants sont nécessaires. :

La segmentation de référence : qui correspond aux résultats attendus de la segmentation. Dans le cas d'images synthétiques, la segmentation de référence est connue. Dans d'autres cas (images naturelles), il relève de l'expert du domaine (médicale ou autre) de définir manuellement cette segmentation de référence, mais le problème concernant l'objectivité et la variabilité des experts demeure.

Une mesure de dissimilitude : entre le résultat de la segmentation et la segmentation de référence Dans ce cas, la qualité d'une segmentation dépend du taux de classification correcte des pixels [Huet and Philipp, 1998].

L'évaluation basée sur des critères non-supervisés : ces critères permettent d'estimer les résultats d'une segmentation sans aucune connaissance à priori de l'image [Zhang, 1996]. L'approche classique dans ce type de méthodes, est basée sur le calcul des mesures statistiques sur les résultats de la segmentation, tels que l'écart type des niveaux de gris ou le contraste de chaque région dans le résultat de la segmentation. Le problème est que la plupart de ces critères ne sont pas adaptés pour des résultats de segmentation de texture [Bartels and Fisher, 1995]. Ceci est un problème important pour l'évaluation d'une segmentation, étant donné que les images naturelles contiennent beaucoup de régions texturées.

Ces critères d'évaluation peuvent être employés pour différentes applications dont trois d'entre elles sont citées dans ce qui suit :

- La comparaison de différents résultats de segmentation pour une image : Il est possible de comparer le comportement des différentes méthodes de segmentation afin de choisir la plus appropriée à un type d'images.
- La facilitation du choix des paramètres d'une méthode de segmentation : Les algorithmes de segmentation d'images ont besoin, généralement, de quelques paramètres d'entrée qui sont habituellement définis par l'utilisateur. Cette tâche parfois arbitraire, peut être automatisée en déterminant les meilleurs paramètres grâce aux critères d'évaluation.

– La définition de nouvelles méthodes de segmentation en optimisant des critères d'évaluation.

Dans les sections suivantes, nous présentons différents critères d'évaluation des méthodes de segmentation en décrivant séparément les deux approches d'évaluation : les critères d'évaluation supervisés et non supervisés.

3.3.1 Critères d'évaluation supervisée

Le principe de cette approche est de mesurer la dissimilitude entre un résultat de segmentation et une segmentation de référence.

Dans le cas des images synthétiques, la segmentation de référence est très fiable et est d'une grande précision. Pour des images naturelles, la segmentation de référence est subjective, vue l'impact du facteur humain.

Dans cette section, on notera I une image de N pixels. Soit I_{seg} l'image segmentée en C_{seg} régions et I_{ref} la segmntation de référence composée de C_{ref} régions. On notera aussi $I_{seg}(i)$ $(i = 1, \ldots, C_{seg})$ les régions composant I_{seg}, et $I_{ref}(i)$ $(i = 1, \ldots, C_{ref})$ les régions composant I_{ref}.

3.3.1.1 Indice de Jaccard

L'indice de Jaccard est un coefficient dont le but et de mesurer la similarité entre deux ensembles de données. Ce coefficient représente le quotient de l'intersection des deux ensembles par leur union. Formellement, il est défini comme suit :

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{3.24}$$

Dans le cas d'une segmentation d'images, l'ensemble A représente une classe dans la segmentation à évaluer I_{seg}, et l'ensemble B est une classe dans la segmentation de référence I_{ref}. Selon le cas, la similarité entre les deux segmentations represente l'ensemble des indices de Jaccard exprimant les taux de correspondance de chaque classe de la segmentation de réference I_{ref} dans I_{seg}.

3.3.1.2 Mesure de Vinet [Vinet, 1991]

Cette mesure calcule la dissimilitude entre deux segmentations. Dans notre cas elle donne la différence entre un résultat de segmentation I_{seg} et une segmentation de référence I_{ref}.Cette mesure consiste à rechercher les couples de régions $(I_{ref}(i), I_{seg}(j))$ es plus similaires. Une matrice de recouvrement est construite à

cet effet :

$$T(I_{ref}, I_{seg}) = t_{i,j} = card\left(I_{ref}(i) \bigcap I_{seg}(j)\right) \ \forall i = 1, \ldots, C_{ref} \ et \ \forall j = 1, \ldots, C_{seg} \tag{3.25}$$

Il s'agit de trouver les couples de régions qui maximisent le recouvrement. Soit C' l'ensemble des couples de régions qui maximisent T. la mesure de Vinet est la suivante :

$$V(I_{ref}, I_{seg}) = N - \sum_{C'} card\left(I_{ref}(i) \bigcap I_{seg}(j)\right) \tag{3.26}$$

La mesure de Vinet est proportionnelle au nombre de pixels n'appartenant pas au recouvrement entre deux classes [Ouadfel, 2006]. Cette mesure n'exige pas un nombre identique de classes dans les deux segmentations.

Cette mesure a été utilisée par Cocquerez (voir [Cocquerez and Philipp, 1995]) pour comparer des segmentations sur des images synthétiques monochromes comportant différents bruits et textures.

3.3.1.3 Mesure de Yasnoff [Yasnoff et al., 1977]

Cette mesure permet de quantifier les erreurs de classification. Elle permet d'évaluer le taux d'erreur d'un pixel mal segmenté p par rapport à la région à laquelle il appartient dans la segmentation de référence. L'erreur entre le résultat d'une segmentation I_{seg} et sa référence I_{ref}, est définie comme suit :

$$Y(I_{seg}, I_{seg}) = \frac{100}{N} \times \sqrt{\sum_{p \in I, p \notin R_p} \min_{q \in R_p} d(p, q)} \tag{3.27}$$

La sommation s'effectue sur les pixels mal segmentés. $\min_{q \in R_p} d(p, q)$ Correspond à la distance entre le pixel mal segmenté p et celui le plus proche q dans R_p.

Cet indice a été utilisé par Zhang [Zhang, 1996] pour comparer des méthodes de seuillage.

3.3.1.4 Mesure de Martin [Martin, 2002]

Définie par D. R. Martin pour évaluer la cohérence entre deux segmentations manuelles d'une même image, cette mesure peut être utilisée pour comparer deux segmentations l'une de référence et l'autre obtenue par un algorithme.

Elle est basée sur deux erreurs calculées en chaque pixel : une erreur de I_{ref} par rapport à I_{seg} et une erreur de I_{seg} par rapport à I_{ref}. Si le pixel s appartient à la région $I_{ref}(j)$ dans la segmentation de référence et à la région $I_{seg}(i)$ dans l'image

résultat. Ces erreurs ont pour valeur :

$$E(s) = \frac{card(I_{ref}(j)/I_{seg}(i))}{card(I_{ref}(j))} \tag{3.28}$$

et

$$E'(s) = \frac{card(I_{ref}(i)/I_{seg}(j))}{card(I_{seg}(i))} \tag{3.29}$$

$E(s)$ vaut 0 si $I_{ref}(j)$ est un sous-ensemble de $I_{seg}(i)$ et vaut 1 si l'intersection entre deux régions est réduite au pixel s.

La dissimilitude entre la segmentation résultat et la segmentation de référence se mesure alors par l'erreur locale de cohérence :

$$LCE(I_{ref}, I_{seg}) = \frac{1}{N} \sum_{s} \min E(s), E'(s) \tag{3.30}$$

Ou par l'erreur globale de cohérence :

$$GCE(I_{ref}, I_{seg}) = \frac{1}{N} \min \sum_{s} E(s), \sum_{s} E'(s) \tag{3.31}$$

Cette dernière mesure est plus sévère que la première mais a le défaut de favoriser une sur-segmentation (ou une sous-segmentation) de toute l'image par rapport à un mélange des deux situations (sur et sous-segmentation selon les zones de l'image).

3.3.1.5 Distance de Hausdorff [Beauchemin et al., 1998]

Ce critère mesure la différence entre deux ensembles de pixels : $E_1 = \{p_1, \ldots, p_n\}$ et $E_1 = \{q_1, \ldots, q_m\}$:

$$H(E_1, E2) = \max(h(E_1, E2), h(E_2, E1)) \tag{3.32}$$

Avec :

$$h(E_1, E2) = \max_{p_i \in E_1} \min_{q_j \in E_2} \|p_i - q_j\| \tag{3.33}$$

Si $h(E_1, E2) = d$ alors tous les pixels appartenant à E_1 sont à une distance inférieure à d des pixels de E_2. Cette mesure est théoriquement très intéressante. Elle donne en effet une bonne mesure de similitude entre les deux images. Cependant, cette méthode est sensible au bruit.

3.3.1.6 Distance de Baddeley [Baddeley, 1992]

La distance de Baddeley s'inspire beaucoup de la distance de Hausdorff. Elle est moins sensible au bruit que cette dernière, et prend en compte la position du pixel dans l'image en plus de son intensité. La distance de Baddeley est définie comme suit :

$$D_B(I_1, I_2) = \left[\frac{1}{card(X)} \sum_{x \in X} |d(x, I_1) - d(x, I_2)|^p \right]^{\frac{1}{p}} \tag{3.34}$$

Avec : I_1 et I_2 correspondent aux ensembles des pixels des contours des deux segmentations à comparer, X est le domaine commun des deux ensembles I_1 et I_2, $d(x, I) = \min_{y \in I} d(x, y)$ et $p \geq 1$.

3.3.2 Critères d'évaluation non supervisés

Il est possible de mesurer la qualité d'une segmentation en employant des critères non supervisés. La plupart de ces critères calculent des statistiques sur chaque région dans le résultat de la segmentation. Pour la segmentation par régions, les divers critères tiennent compte de l'uniformité intra-région et du contraste inter-régions.

3.3.2.1 Critères d'uniformité intra-région

L'un des critères les plus intuitifs pouvant mesurer la qualité d'un résultat de segmentation est l'uniformité intra-région. Nazif et Levine [Levine and Nazif, 1985] ont défini un critère qui mesure l'uniformité du niveau de gris dans une région, basé sur la variance de ce niveau de gris. Ce critère est défini comme suit :

$$L(I) = 1 - \sum_{k=1}^{NR} \frac{\sum_{p_i \in R_k} [g_I(p_i) - \sum_{p_j \in R_k} g_I(p_j)]^2}{card(I) \times (\max_{p_i \in R_k} g_I(p_i) - \min_{p_i \in R_k} g_I(p_i))} \tag{3.35}$$

Oú :

- $g_I(p_i)$ correspond au niveau de gris du pixel p_i dans l'image I ;
- R_k représente la région k dans l'image segmentée ;
- $card(I)$ correspond au nombre de pixels dans l'image I.

Une autre variante du critère de Levine et Nazif, est celle proposée par Sahoo [Sahoo et al., 1988]. Ce critère est défini comme suit :

$$S(I) = 1 - \frac{L_2(I)}{card(I)} \tag{3.36}$$

Avec :

$$L_2(I) = \sum_{k=1}^{NR} \sum_{p_i \in R_k} \left[g_I(p_i) - \frac{1}{card(R_k)} \sum_{p_i \in R_k} g_I(p_i) \right]^2 \tag{3.37}$$

3.3.2.2 Critères de contraste inter-régions

Ce type de mesure est complémentaire à l'uniformité intra-classe. Levine et Nazif [Levine and Nazif, 1985] ont défini un critère de mesure basé sur le contraste pour mesurer la dissimilitude entre tout couple de régions R_i et R_i du résultat de la segmentation. Ce critère de mesure est défini comme suit :

$$C = \frac{\sum_{i=1}^{NR} w_i \sum_{j=1}^{NR} \frac{l_{i,j}}{l_i} \frac{|m_i - m_j|}{|m_i + m_j|}}{\sum_{i=1}^{NR} w_i} \tag{3.38}$$

Oú :

- w_i est un poids associé à chaque région (qui peut être sa surface) ;
- l_i est la longueur du contours de la région R_i ;
- $l_{i,j}$ est la longueur de la frontière entre les deux région R_i etR_j.

Ce critère a pour avantage de pénaliser la sur-segmentation.

3.3.2.3 Critères de contraste inter-intra régions

Dans cette catégorie de critères d'évaluation, le résultat de la segmentation est mesuré selon les deux critères définis précédemment : les critères de contraste inter-régions et les critères de contraste intra-région.

Critère de Rosenberg : Parmi les critères d'évaluation de cette catégorie on peut citer celui de Rosenberg [Rosenberg, 1999] qui permet d'estimer l'homogénéité des régions d'un résultat de segmentation de manière à maximiser l'homogénéité intra-classe et le contraste inter-classes. Il repose sur deux critères heuristiques suggérés par Haralick et Shapiro [Haralick and Shapiro, 1985] pour évaluer les résultats de segmentation :

- les régions doivent être homogènes ;
- les régions adjacentes doivent présenter des valeurs significativement différentes pour les caractéristiques utilisées.

Le critère de Rosenberg est défini comme suit :

$$R(I_{seg}) = \frac{\overline{D}(I_{seg}) - \underline{D}(I_{seg})}{2} \tag{3.39}$$

$\overline{D}(I_{seg})$ Correspond à la disparité intra-région, qui quantifie l'homogénéité de chacune des régions de l'image I_{seg}.

$$\overline{D}(I_{seg}) = \frac{1}{NR}\sum_{k=1}^{NR}\frac{card(R_k)}{card(I)}\overline{D}(R_k) \tag{3.40}$$

$\overline{D}(R_i)$ correspond à la disparité intra-classe de la région i. Ce qui est défini par l'écart-type des intensités pour une région uniforme et par un ensemble d'attributs de texture pour une région texturée. Pour les régions uniformes la disparité intra-classe d'une région est définie par [Ouadfel, 2006] :

$$\overline{D}(R_i) = \frac{2}{255}\sqrt{\frac{1}{card(R_i)}\sum_{p_i\in R_i} g_I^2(p_i) - \frac{1}{card(R_i)^2}\left(\sum_{p_i\in R_i} g_I(p_i)\right)^2} \tag{3.41}$$

$\underline{D}(I_{seg})$ correspond à la disparité inter-classes, qui mesure la disparité globale de chacune des régions de l'image I_{seg}. Elle est définie par la fonction suivante :

$$\underline{D}(R_k) = \frac{1}{q^{(k)}}\sum_{j=1}^{q^{(k)}}\underline{D}(R_k, R_j) \tag{3.42}$$

Avec $q^{(k)}$ le nombre de régions R_j voisines à R_k. La disparité $\underline{D}(R_k, R_j)$ se calcule par :

$$\underline{D}(R_k, R_j) = \frac{|\overline{g}(R_k) - \overline{g}(R_j)|}{NG} \tag{3.43}$$

$\overline{g}(R_i)$ est la moyenne des niveaux de gris de la région, et NG est le nombre de niveaux de gris dans l'image.

Critère de Liu et Yang [Liu and Yang, 1994] : La mesure proposée par Lui et Yang incorpore les propriétés suivantes :

- les régions doivent être uniformes et homogènes ;
- l'intérieur des régions doit être simple et sans trop de « trous » ;
- les régions adjacentes doivent présenter des valeurs significativement différentes pour les caractéristiques d'uniformité.

Cette mesure est définie par la fonction suivante :

$$L(I_{seg}) = \frac{1}{1000\times card(I)}\sqrt{NR}\sum_{i=1}^{NR}\frac{e_i^2}{\sqrt{card(R_i)}} \tag{3.44}$$

Oú est la somme des distances euclidiennes entre les niveaux de gris des pixels de la région R_i et le niveau de gris moyen de cette région.

Le terme $\sqrt{NR}$ pénalise des résultats comportant plusieurs petites régions (sur-segmentation) [Ouadfel, 2006]. Le calcul de la somme des erreurs intra-région pénalise les résultats de segmentation comportant des régions non-homogènes. Une bonne segmentation minimise la fonction $L(I_{seg})$.

Critère de Borsotti[Borsotti et al., 1998] : Ce critère est proposé par Borsotti afin d'améliorer la fonction d'évaluation établie par Liu et Yang, qui présente certaines limitations. En effet, quand un résultat de segmentation présente un grand nombre de « petites » régions (sur-segmentation), le nombre total de régions est donc élevé, par conséquent, le résultat de la segmentation est pénalisé par la fonction de Liu et Yang. Cependant, l'erreur intra-classe sera proche de zéro, et $L(I_{seg})$ sera aussi proche de zéro, ce qui suppose une très bonne segmentation, ce qui est incorrecte.

Borsotti et al proposent par rapport à Liu et Yang de pénaliser les résultats de segmentation avec un grand nombre de « petites » régions ainsi que les résultats avec des régions non-homogènes. La mesure de Borsotti est définie par :

$$B(I_{seg}) = \frac{1}{1000 \times card(I_{seg})} \sqrt{NR} \sum_{i=1}^{NR} \left[\frac{e_i^2}{1 + log(card(R_i)} + \left(\frac{f(card(R_i))}{card(R_i)} \right)^2 \right] \tag{3.45}$$

Oú $f(card(R_i))$ correspond au nombre de régions ayant la même aire $card(R_i)$.

Le premier terme de la somme pénalise les régions non-homogènes. Le second terme pénalise les régions dont l'aire $card(R_i)$ est semblable à beaucoup d'autres régions. Une bonne segmentation minimise la fonction $B(I_{seg})$.

3.4 Conclusion

Comme nous l'avons vu dans ce chapitre, le problème de la segmentation d'image peut être considéré comme étant un cas particulier du problème de partitionnement des données. Ce dernier étant un problème NP-complet il peut être résolu par des métaheuristiques telles les méthodes que nous décrirons dans le chapitre suivant.

Chapitre 4

Introduction aux méthodes d'optimisation

4.1 Introduction

Les problèmes qui doivent être résolus par les ingénieurs et les décideurs dans divers domaines techniques sont de plus en plus complexes, et trouver une solution à un problème se traduit souvent par la résolution d'un problème d'optimisation. On appelle problème d'optimisation un problème dont la résolution consiste en la minimisation ou la maximisation d'une fonction objectif (fonction de coût) par rapport à ses paramètres.

De nouvelles méthodes, appelées métaheuristiques, sont apparues à partir des années 1980, avec pour objectif commun de résoudre au mieux les problèmes dits d'optimisation. Parmi ces méthodes nous citons :

- la méthode du recuit simulé ;
- les algorithmes évolutionnaires ;
- la méthode de recherche taboue ;
- les algorithmes de colonies de fourmis ;
- etc ...

Les métaheuristiques sont des méthodes génériques pouvant être appliquées sur une large gamme de problèmes. Dans ce qui suit, nous présentons quelques unes de ces méthodes.

4.2 Algorithmes évolutionnaires

Le premier ouvrage traitant de la Théorie de l'évolution a été publié en 1859 par Charles Darwin sous le titre « L'origine des espèces au moyen de la sélection

naturelle »

Cet ouvrage énonce que, l'évolution des systèmes vivants au cours des générations s'opère en deux étapes : la sélection naturelle et la reproduction :

- La sélection naturelle est le mécanisme de base qui régit l'existence des populations : il entraîne la mort des individus les plus faibles et la survie des individus les mieux adaptés à l'environnement ;
- La reproduction, résultat d'une succession de mutations et de combinaisons, entraîne des modifications multiples sur les individus d'une population aboutissant ainsi à une grande diversité entre deux populations d'une même espèce.

D'autre part, la littérature scientifique est également riche de nombreux travaux dont ceux de Mandel portant sur la transmission des caractères héréditaires d'une génération à une autre dans le cadre d'une reproduction sexuée.

Cette théorie a pour fondement que les caractères héréditaires sont localisés dans le génome qui constitue le patrimoine génétique de chaque individu et qui est défini comme un ensemble de gênes caractéristiques de l'espèce.

La reproduction sexuée est basée sur deux principes : Le croisement et la mutation génétique. Le croisement est caractérisé par la combinaison du patrimoine génétique des parents engendrant, ainsi, des individus distincts.

La mutation est caractérisée par la modification spontanée de quelques gènes durant la phase du croisement.

La découverte de l'ADN en 1953 par Francis H. C. Crick et James D. Watson, a fait suite aux travaux de Mandel, et a révolutionné le domaine de la génétique.

Ces théories ont fortement inspiré les chercheurs informaticiens qui ont adapté les propriétés de croisement et de mutation sur des modèles artificiels applicables au domaine de l'optimisation.

Parmi ces modèles on trouve une classe d'algorithmes regroupés sous le nom générique d'algorithmes évolutionnaires (AE), apparus dans les années 70 avec les travaux d'Ingo Rechenberg et ceux de Hans-Paul Schwefel [Schwefel, 1981].

Les algorithmes génétiques (AG) sont la branche des AE la plus connue et la plus utilisée.

Les algorithmes évolutionnaires se différencient par leur façon de coder les individus (donc modéliser le problème à résoudre) et par leur façon de faire évoluer la population.

4.2.1 Les algorithmes génétiques

Les algorithmes génétiques (AG) sont des méthodes de programmation inspirées de la théorie de l'évolution de Darwin et de la recombinaison génétique de Mandel

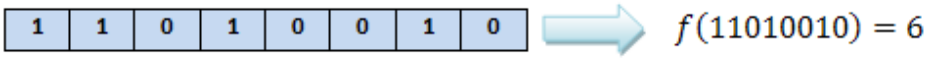

FIG. 4.1 – Codage de l'information

pour rechercher une « bonne » solution à un problème d'optimisation. Ces méthodes ont été crées par John Holland, [Holland, 1975], puis développées par d'autres chercheurs.

Le principe de base des AG est de faire évoluer une population d'individus afin d'obtenir des solutions optimisant de mieux en mieux la fonction de coût. Les étapes d'un algorithme génétique peuvent être résumées comme suit :

1. Choix d'un codage approprié pour les individus de la population. Ce codage doit être complet et capable de coder toutes les solutions possibles. Dans un premier temps, Holland a utilisé un codage sous forme de chaîne de bits de longueur fixe afin de maintenir le plus possible l'analogie avec la structure protéinique de l'ADN. Le codage binaire possède l'avantage de fournir un langage quasi-universel, indépendant du problème traité. Le codage binaire d'un individu est appelé « chromosome » et ses éléments sont appelés « gènes ». Plus récemment, d'autres types de codage sont apparus pour pallier à la complexité de mise en œuvre du codage binaire [Goldberg, 1989] ;
2. Construction de la population initiale d'individus ;
3. Association d'un coût à chaque individu de la population, calculé à l'aide d'une fonction d'évaluation (fitness), qui mesure le degré d'adaptation de l'individu à l'objectif visé. La figure 4.1 donne un exemple de codage d'un chromosome sur 8 bits. L'évaluation de cet individu consiste à transformer la chaîne 0/1 en une valeur réelle ;
4. Evolution progressive, par générations successives de la population. Au cours des générations, l'objectif est d'améliorer globalement la performance des individus par l'application des opérateurs génétiques : sélection, croisement et mutation.

4.2.1.1 Sélection

La sélection consiste à choisir des individus qui garantiront les meilleures solutions pour la suite du processus d'évolution.

Il existe plusieurs méthodes pour sélectionner les individus. Les méthodes les plus couramment utilisées sont : la sélection par roue de loterie, la sélection par tournois

Individu	Fitness	Fitness relative
1	22.5	0.15
2	37.5	0.25
3	18.75	0.125
4	18.75	0.125
5	7.5	0.05
6	45	0.3

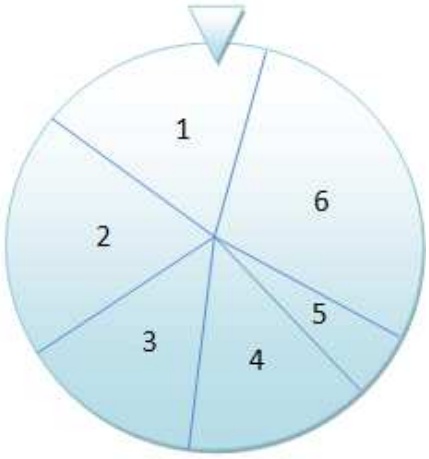

FIG. 4.2 – Roue de la loterie

et l'élitisme [Developez.com, 2007].

A. **La sélection par la roue de loterie**

Comme son nom l'indique, ce système est inspiré de la roulette utilisée dans les jeux de hasard. Dans cette méthode, à chacun des individus de la population (chromosomes) est attribué un secteur sur la roue, l'angle du secteur étant proportionnel à l'individu qu'il représente : l'individu dont la fonction d'adaptation est la plus grande sera représenté par un secteur plus large sur la roue et inversement. On fait tourner la roue et on obtient un individu. Les tirages des individus sont ainsi pondérés par leur qualité. Par conséquent, les meilleurs individus ont plus de chance d'être croisés et de participer à l'amélioration de la population.

B. **La sélection par tournoi**

Le principe de sélection par tournoi est d'augmenter la chance des individus de moindre qualité de participer à l'amélioration de la population. L'idée consiste à choisir aléatoirement des individus de la population, le vainqueur de cette partie du tournoi étant l'individu de meilleure qualité. De cette façon, on peut faire plusieurs parties, et élire les meilleurs individus (qui ne sont pas forcément les meilleurs de la population).

C. **Élitisme**

Cette méthode de sélection permet de mettre en avant les meilleurs individus de la population. Ceux sont donc les individus les plus prometteurs qui vont participer à l'amélioration de la population. Cette méthode a l'avantage de

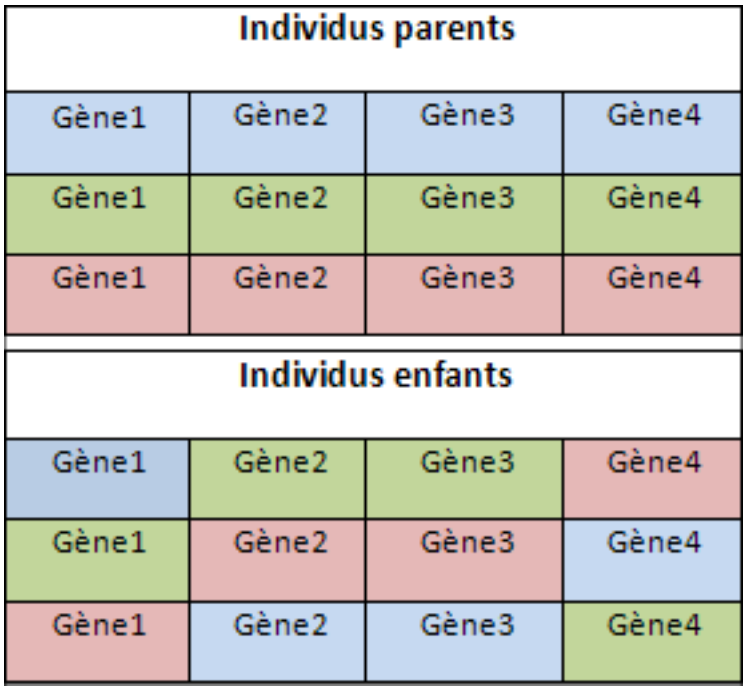

FIG. 4.3 – Croisement en plusieurs points

permettre une convergence rapide des solutions, mais au détriment de la diversité des individus. Cette méthode est extrêmement sensible à la présence d'optimums locaux.

4.2.1.2 Croisement

Cette étape porte sur la création de nouveaux individus à partir de ceux choisis au cours de la sélection.

La méthode de croisement la plus utilisée est le croisement en plusieurs points, mais il existe d'autres méthodes, parmi elles le croisement uniforme.

A. **Croisement en plusieurs points**

   Dans cette méthode, les individus choisis pour le croisement sont découpés en plusieurs morceaux. Chaque individu résultant sera construit à partir des différentes parties (gènes) des individus racine (Voir la figure 4.3).

B. **Croisement uniforme**

   Dans cette méthode, un masque, constitué de valeurs binaires, sera utilisé. Les deux individus (chromosomes) initiaux constitueront les deux chromosomes résultants suivant les valeurs du masque (Voir l'algorithme 2) .

Algorithme 2 Algorithme Croisement Uniforme

si le *ieme* bit du masque est à 0 **alors**

Le i^{eme} gène du premier individu de base sera le i^{eme} gène du premier individu résultant

Le i^{eme} gène du deuxième individu de base sera le i^{eme} gène du deuxième individu résultant

sinon

Le i^{eme} gène du premier individu de base sera le i^{eme} gène du deuxième individu résultant

Le i^{eme} gène du deuxième individu de base sera le i^{eme} gène du premier individu résultant

finsi

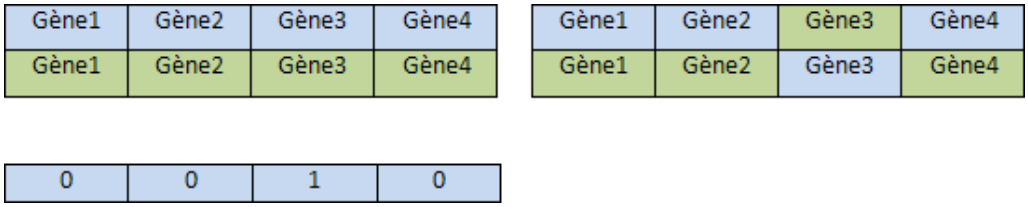

FIG. 4.4 – Croisement uniforme

4.2.1.3 Mutation

La mutation consiste à modifier aléatoirement les valeurs de certains gènes d'un individu. L'opérateur de mutation agit sur un seul individu choisi avec une probabilité P_m, Il permet ainsi de créer une diversité dans la population et d'éviter une convergence prématurée de l'algorithme.

La figure 4.5 montre un exemple de mutation sur un chromosome codé sur 8 bits.

La figure 4.6 résume les différentes étapes d'un algorithme génétique.

Les AG utilisent le principe de « compétition entre individus », de la théorie de l'évolution naturelle, pour créer des systèmes pouvant s'adapter à de nombreuses situations. Cela permet de résoudre des problèmes d'optimisation en explorant « efficacement » l'ensemble des solutions. Les algorithmes génétiques recherchent une

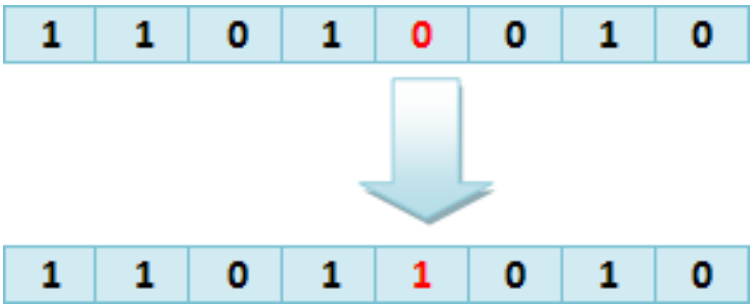

FIG. 4.5 – Opérateur de mutation sur un chromosome de 8 bits

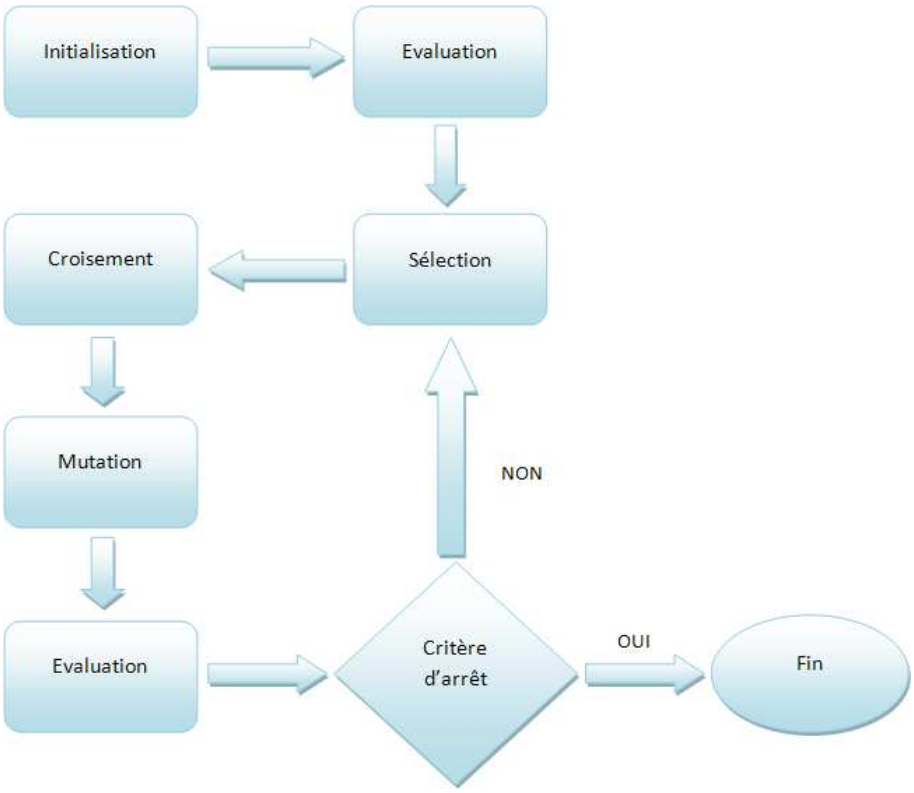

FIG. 4.6 – Structure d'un algorithme génétique [Ouadfel, 2006].

«bonne » solution à un problème parmi un ensemble de solutions en tirant parti de l'historique de cette recherche afin de favoriser l'émergence des solutions les mieux adaptées. Les algorithmes génétiques ont fait leur preuve dans des domaines très variés. Cependant et afin de les appliquer à un problème particulier, il est nécessaire de définir correctement les points suivants [Ouadfel, 2006] :

- La fonction d'évaluation ;
- Le codage des solutions ;
- Les opérateurs de mutation et de croisement.

Toutefois, la recherche d'une solution par les AG ne garantit pas l'obtention d'une solution optimale. Une population initiale mal choisie ou une convergence trop rapide vers un optimum local peut bloquer le processus de résolution. Il n'existe pas de méthodes qui permettent de fixer :

- La taille optimale de la population ;
- Le meilleur codage ;
- Le taux de mutation et de croisement utilisé.

Ces paramètres de fonctionnement dépendent du problème à résoudre, et sont dans la majorité des cas fixés de façon empirique.

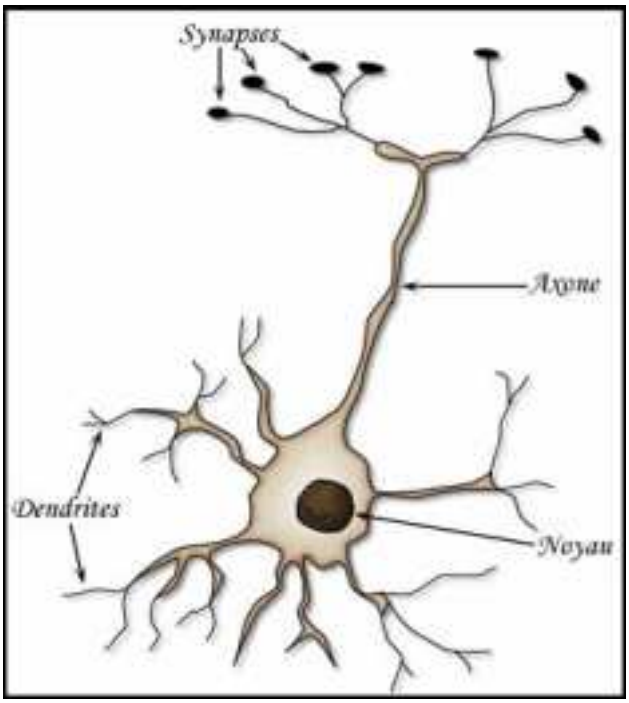

FIG. 4.7 – Morphologie d'un neurone [Ouadfel, 2006]

4.3 Réseaux de neurones

De la même manière que les algorithmes évolutionnaire ont été inspirés de l'évolution naturelle des espèces, le domaine des réseaux de neurones (RN) porte sur une simulation du fonctionnement du système nerveux humain.

Les réseaux de neurones formels sont des modèles théoriques de traitement de l'information inspirés des observations relatives au fonctionnement des neurones biologiques et du cortex cérébral. Par analogie aux neurones biologiques, les neurones artificiels ont pour but de reproduire des raisonnements « intelligents ». Ces neurones peuvent adopter certaines qualités habituellement propres à leurs équivalents biologiques, entre autres, la généralisation, l'évolutivité, et une certaine forme de déduction.

4.3.1 Les neurones biologiques

Les cellules nerveuses, appelées neurones, constituent la base du système nerveux central qui se compose d'environ 10^{12} neurones. Le neurone se compose de quatre parties essentielles (Figure 4.7) :

Un corps cellulaire : qui contient le noyau et a pour rôle d'effectuer les transformations biochimiques nécessaires à la synthèse des éléments assurant la vie du neurone ;

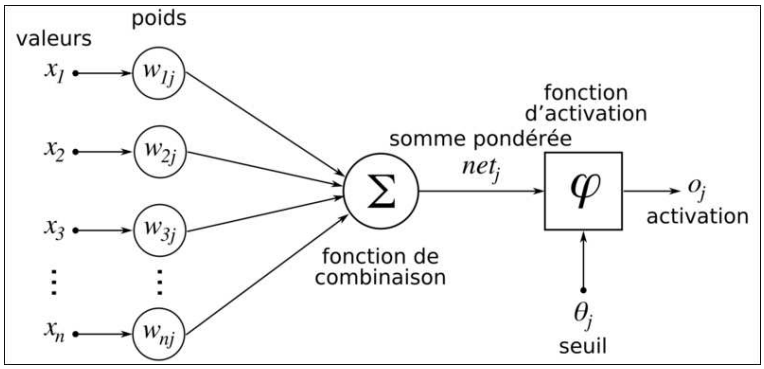

FIG. 4.8 – La structure générale d'un neurone formel

Les dendrites : qui sont des ramifications du corps cellulaire. Elles permettent au neurone de capter les signaux parvenant de l'extérieur ;

L'axone : généralement plus long que les dendrites, il se ramifie à son extrémité à partir de laquelle il communique avec les autres neurones. Il sert de moyen de transport pour les signaux émis par le neurone ;

Les synapses : qui réalisent la connexion entre l'axone émetteur et les dendrites réceptrices.

Le neurone biologique reçoit des impulsions de neurones voisins avec lesquels il est connecté au travers des synapses. Les influx nerveux transmis par les dendrites sont sommés. Si la sommation dépasse un seuil, le neurone répond par un influx nerveux qui se propage le long de son axone. Si la sommation est inférieure au seuil, le neurone reste inactif. Les premières cellules qui alimentent le réseau sont constituées par des capteurs (cellules sensorielles) comme les cellules de la rétine de l'œil, par exemple.

4.3.2 Du neurone biologique au neurone formel

Un réseau de neurones (RN) est un ensemble de neurones formels connectés entre eux. Il est, dans sa structure, inspiré du système nerveux humain. Comme dans ce dernier, les neurones formels constituant le RN sont interconnectés entre eux, de manière à ce que les signaux sortants des neurones deviennent des signaux entrant d'autres.

Un neurone artificiel est un automate à deux états « actif » et « inactif » : un neurone devient « actif » si la somme de l'ensemble des signaux entrants pondérés dépasse un certain seuil, il devient « inactif » sinon.

Un réseau de neurones est caractérisé, principalement, par sa topologie (figure 4.9), et par sa fonction d'activation (figure 4.10).

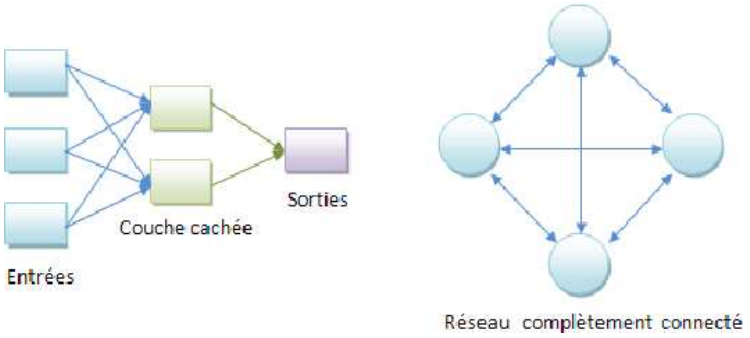

FIG. 4.9 – Quelques Topologies de Réseau de Neurones

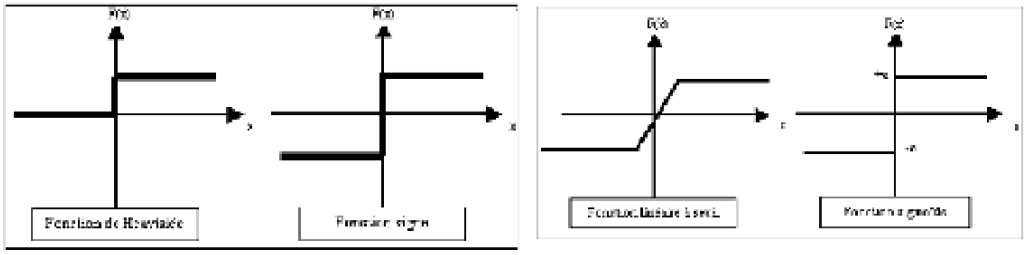

FIG. 4.10 – Quelques Fonctions d'activation [Ouadfel, 2006]

D'une façon plus générale, on définit un neurone formel par les cinq paramètres suivants :

1. Le type des entrées (booléenne ou réelle) ;
2. La fonction d'entrée totale, définissant le prétraitement effectué sur les entrées ;
3. La fonction de seuillage (appelée aussi fonction d'activation) du neurone définissant son état interne en fonction de la somme pondérée de ses entrées ;
4. La fonction de sortie calculant la sortie du neurone en fonction de son état d'activation ;
5. Le type des sorties du neurone.

4.4 Optimisation par essaim de particules

Les algorithmes à essaim de particules, sont inspirés du comportement de déplacement collectif observé chez certains animaux sociaux tels les oiseaux migrateurs [Kennedy and Eberhart, 2001].

En effet, ces animaux se déplacent avec une harmonie étonnante, se devisent pour éviter des obstacles ou des prédateurs, puis se rassemblent à nouveau pour former le groupe initial et ce, grâce à des règles assez simples telles que :

– Rester proche des autres individus ;

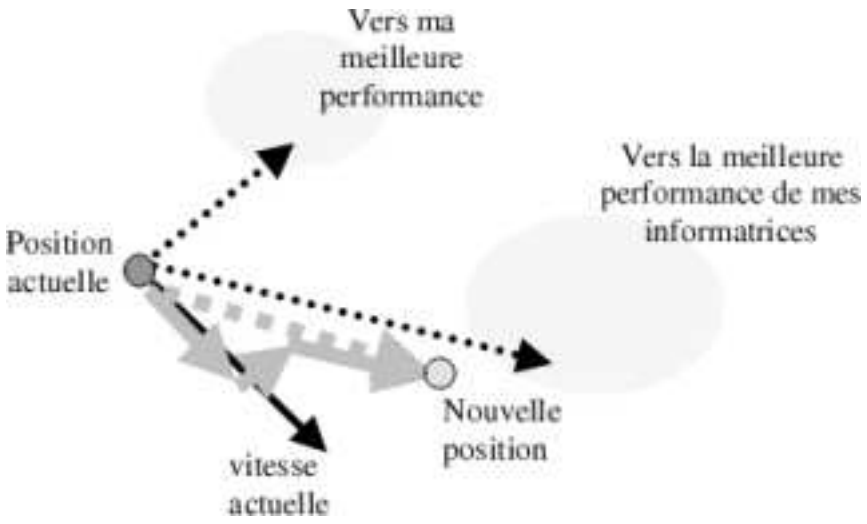

FIG. 4.11 – Schéma de déplacement d'une particule dans le voisinage, chaque particule combine trois mouvement : suivre sa vitesse propre, revenir vers sa meilleure performance, aller vers la meilleure performance de ses voisines [Clerc, 2002].

- Aller dans la même direction que le groupe ;
- Aller à la même vitesse que le groupe.

Dans l'algorithme à essaim de particules, les individus sont appelés particules et la population est l'essaim [Ouadfel, 2006].

4.4.1 Algorithme d'optimisation par essaim de particules

Dans l'algorithme d'optimisation par essaim de particules, un essaim d'individus (particules) parcourt l'espace de recherche. Chaque particule représente une solution candidate au problème d'optimisation. La position d'une particule est influencée par sa meilleure position et la position de la meilleure particule dans son voisinage.

Quand le voisinage d'une particule est l'essaim entier, la meilleure particule voisine est la meilleure particule globale. L'algorithme utilisant cette propriété est désigné sous le nom de PSO *gbest* (global best particle).

Quand des voisinages plus restreints sont employés, l'algorithme est désigné sous le nom de *lbest* PSO (local best particle).

Le coût de chaque particule est mesuré en utilisant une fonction de fitness définie selon le problème d'optimisation. Chaque particule de l'essaim est représentée par les caractéristiques suivantes :

- x_i la position courante de la particule i.
- v_i la vitesse courante de la particule i.
- y_i la meilleure position de la particule i

– $\hat{y}_i$ la meilleure position du voisinage d'une particule.

La mise à jour de la meilleure position d'une particule est effectuée grâce à la fonction de fitness f, qui dénote le coût d'une solution candidate. La mise à jour de la meilleure position d'une particule à l'instant t est réalisée par :

$$y_i(t) = \begin{cases} y_i(t-1) & sif(x_i(t)) \geq f(y_i(t-1)), \\ x_i(t-1) & sif(x_i(t)) < f(y_i(t-1)). \end{cases} \tag{4.1}$$

Dans le modèle de l'optimisation par essaim de particules gbest, dans lequel la meilleure particule de l'essaim est choisie parmi toutes les meilleures positions des particules de tout l'essaim, la meilleure particule $\hat{y}$ est calculée comme suit :

$$\hat{y}(t) = \{y_i(t),\ tq\ f(y_i(t)) = \min\{f(y_0(t), \ldots, f(y_s(t)))\}\} \tag{4.2}$$

Avec s qui représente la taille de l'essaim. La vitesse d'une particule est mise à jour pour chaque élément de la particule à l'aide de l'équation suivante :

$$v_{i,j}(t+1) = wv_{i,j}(t) + c_1 r_{1,j}(t)(y_{i,j}(t) - x_{i,j}(t)) + c_2 r_{2,j}(\hat{y}_i(t) - x_{i,j}(t)) \tag{4.3}$$

Avec w qui représente l'inertie, c_1 et c_2 sont des constantes d'accélération, $r_{1,j}(t)$ et $r_{2,j}(t)$ suivent une loi uniforme de moyenne nulle et d'écart-type égal à 1.

La position d'une particule est mise à jour par l'équation suivante :

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{4.4}$$

Dans l'algorithme d'optimisation par essaim de particules, les particules se déplacent dans l'espace des solutions jusqu'à atteindre un critère d'arrêt, en l'occurrence, le nombre d'itérations spécifié. La qualité d'une particule est mesurée par une fonction de fitness qui est définie selon le problème à résoudre. Le schéma général de l'algorithme d'optimisation par essaim de particules est montré dans l'algorithme 3.

4.5 Algorithmes de fourmis artificielles

Ces dernières années, beaucoup de thèses de recherche en informatique, relevant du domaine de l'optimisation, se sont inspirées des études éthologiques sur les insectes, et plus particulièrement les fourmis.

« Les sociétés d'insectes nous proposent un modèle de fonctionnement bien dif-

Algorithme 3 Algorithme d'optimisation par essaim de particules

pour chaque particule $i \in \{1, \ldots, S\}$ **faire**
  Initialiser aléatoirement x_i
  Initialiser aléatoirement v_i
  Poser $y_i = x_i$
fin pour
répéter
  pour chaque particule $i \in \{1, \ldots, S\}$ **faire**
    Evaluer la fitness d'une particule i, $f(x_i)$
    Mettre à jour la meilleure position d'une particule y_i
    Mettre à jour la meilleure position de l'essaim $\hat{y}$
    Mettre à jour la vitesse d'une particule $v_{i,j}$ $\forall j \in \{1, \ldots, n_d\}$
    Mettre à jour la position courante de la particule, x_i
  fin pour
jusqu'à atteindre un critère de terminaison

férent du modèle humain : un modèle décentralisé, fondé sur la coopération d'unités autonomes au comportement relativement simple et probabiliste, qui sont distribuées dans l'environnement et ne disposent que d'informations locales. » [Deneubourg et al., 1991].

Les petites créatures faibles que sont les fourmis, arrivent à résoudre collectivement beaucoup de problèmes quotidiens pour la survie de leur espèce. Ces problèmes qui peuvent être aussi divers que la collecte de nourriture, la construction du nid ou encore le tri des cadavres, sont résolus par les fourmis de manière optimale, ce qui a suscité l'intérêt des chercheurs pour ces créatures infimes afin de résoudre des problèmes d'optimisation. Une nouvelle classe d'algorithme à vu le jour : « algorithme de fourmis artificielles » [Ouadfel, 2006]. Les chercheurs se sont focalisés sur deux comportements principaux : l'optimisation de chemin et le tri des couvains.

Le premier comportement appelé aussi fourragement permet aux fourmis de retrouver le plus court chemin entre le nid et l'endroit où se trouve la nourriture et ce par un système de marquage par phéromone. Ce comportement a été modélisé pour résoudre de nombreux problèmes d'optimisation, ce qui à donné naissance à une classe d'algorithmes connue sous le nom de « Ant Colony Optimisation » (ACO). Le deuxième comportement, qui est le tri des cadavres, a été modélisé pour résoudre des problèmes liés à la classification.

4.5.1 Algorithmes inspirés des fourmis naturelles pour les problèmes d'optimisation

Les fourmis sont des insectes vivants en société dans des nids appelés fourmilières, ces petites créatures, rappelons le, sont capables de résoudre d'une manière

optimales plusieurs problèmes quotidiens en s'unissant et en travaillant ensemble. On s'intéresse à présent à deux types de problèmes que peuvent résoudre les fourmis : la recherche de nourriture et le tri de couvains. La recherche de nourriture et l'un des problèmes principaux pour la survie des espèces, que ce soit dans le monde des insectes ou dans celui des mammifères. Chaque espèce a adopté (instinctivement) des stratégies pour se procurer de la nourriture. La stratégie de recherche de nourriture chez les fourmis (appelée aussi fourragement) a été appliquée pour la première fois pour traiter le problème du voyageur de commerce par Dorigo et ces collègues [Colorni et al., 1991]. Par la suite, cette stratégie a été adoptée par les chercheurs pour résoudre d'autres problèmes d'optimisation.

Les fourmis utilisent *la stigmergie* pour mener à bien la tâche de recherche de nourriture. En se déplaçant du nid à la recherche de la source de nourriture, les fourmis laissent derrière elles une substance volatile appelée « phéromone » pour tracer le chemin à suivre par les autres fourmis. La phéromone, de par sa volatilité, tend à faire oublier les mauvais chemins et à renforcer l'empreinte du plus court chemin vers la nourriture. Pour bien comprendre ce comportement, des chercheurs tels que Doneubourg et Goss[Deneubourg et al., 1990] ont mené des expériences (tel le pont binaire et le pont avec obstacle) qui ont abouti à un ensemble d'algorithmes d'optimisation.

D'autre part, le tri des couvains a inspiré de nombreux travaux dont ceux de Deneubourg qui se basaient sur une colonie de fourmis artificielles se déplaçant aléatoirement sur une grille rectangulaire et déplaçant des objets sur cette grille dans le but de les regrouper selon un critère de similarité (De la même manière que les fourmis réelles trient les différents éléments du couvain).

4.5.2 Optimisation par colonies de fourmis (Ant Colony Optimisation) (ACO)

La stratégie du fourragement (recherche de nourriture) fut adaptée pour la première fois à un problème d'optimisation (problème du voyageur de commerce) par Marco Dorigo [Dorigo et al., 1991].

La réussite de cette première modélisation a donné naissance à d'autres algorithmes basés sur la recherche de nourriture chez les fourmis.

Les algorithmes d'optimisation s'inspirants de la stratégie de recherche de nourriture chez les fourmis, sont connus sous le nom «Optimisation par Colonies de fourmis » ou « Ant Colony Optimisation ». Dans l'ACO l'espace des solutions est modélisé par le voisinage du nid, chaque solution est associée à une source de

nourriture dont la qualité est représentée par une fonction objective. Chaque fourmi représente un processus aléatoire contribuant à la construction des solutions. La construction des solutions est biaisée par une phéromone qui représente la mémoire des fourmis sur les solutions antérieures. Elle est régulièrement mise à jour de manière à prendre en considération l'évaporation de la substance (disparition des mauvais chemins).

La résolution par ACO requière une bonne formalisation des différents éléments du problème à optimiser.

Il existe plusieurs algorithmes basés sur l'ACO, Nous citons trois d'entre eux :

Ant System (AS) : C'est le premier algorithme inspiré du fourragement chez les fourmis appliqué au problème du voyageur de commerce ;

Ant Colony System (ACS) : Introduit par Dorigo. Il est une est l'amélioration de l'algorithme AS.

Max-Min Ant System (MMAS) : Introduit par Stutzle et Hoos, il présente une amélioration du ACS dans les étapes d'initialisation et de mise à jour de phéromones [Stuzle and Hoos, 1999].

4.5.3 Classification par fourmis artificielles

De la même façon que les algorithmes regroupés sous la catégorie de *l'optimisation par colonie de fourmis*, qui s'inspirent de la stratégie de recherche de nourriture chez les fourmis, la classification par fourmis artificielles s'inspire du tri collectif de couvains ou la construction de cimetières. Les premiers travaux dans ce domaine ont été ceux de Deneubourg et son équipe en 1990 [Deneubourg et al., 1990].

La classification par fourmis artificielles se base sur un ensemble d'automates se déplaçant aléatoirement sur une grille rectangulaire et qui sont capables de ramasser et déposer des objets présents sur la grille, dans le but d'obtenir une classification des objets selon un critère donné.

De nombreux algorithmes portant sur la classification par fourmis artificielles ont été développés. Nous décrivons quelques uns d'entres eux dans ce qui suit.

4.5.3.1 Algorithme L.F (Lumer et Faieta)

Dans cet algorithme de classification, les objets sont disposés sur une grille rectangulaire, chaque fourmi artificielle est capable de percevoir une région R_s de $s \times s$ cases, la dissimilitude f entre les objets et mesurée par une distance euclidienne entre les attributs caractérisant les objets.

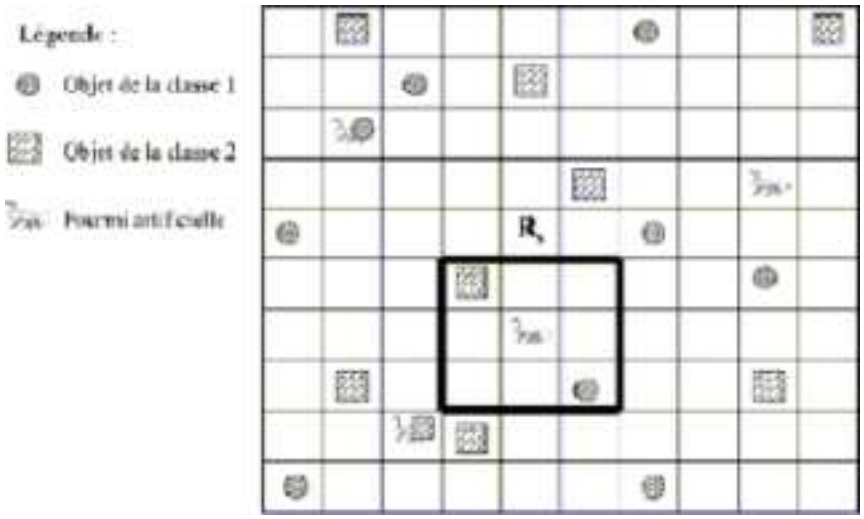

FIG. 4.12 – Grille de classification de Lumer et Faieta [Labroche, 2003].

La figure représente la grille utilisée dans l'algorithme Lumer et Faieta, schématisant deux objets et la région perçue par une fourmi artificielle.

Chaque fourmi a une probabilité P_p de ramasser un objet, si elle ne transporte pas d'objets. Elle a une probabilité P_d de déposer un objet dans une case vide.

L'algorithme L.F se présente comme suit :

Algorithme 4 Algorithme L.F (Lumer et Faita)

Placer aléatoirement les N objets $O_1, O_2, \ldots, O_N$ sur une grille G
pour $T = 1$ à T_{max} **faire**
  pour tout $a_j \in \{a_1, \ldots, a_A\}$ **faire**
    si la fourmi a_j ne transporte pas d'objets et $r(O_i) = r(a_j)$ **alors**
      calculer $f(O_i)$ et $P_p(O_i)$
      la fourmi a_j ramasse l'objet O_i suivant la probabilité $P_p(O_i)$
    sinon
      si la fourmi a_j transporte l'objet O_i et la case $r(a_j)$ est vide **alors**
        calculer $f(O_i)$ et $P_p(O_i)$
        la fourmi a_j depose l'objet O_i sur la case $r(a_j)$ avec une probabilité $P_d(O_i)$
      finsi
    finsi
    Déplacer la fourmi a_j sur une case voisine non occupée par une autre fourmi
  fin pour
fin pour
Retourner l'emplacement des objets sur la grille

4.5.3.2 Algorithme AntClass

Cet algorithme est présenté par Monmarché en 2000[Monmarché, 2000]. Il s'est inspiré de l'algorithme LF, avec quelques modifications de ce dernier.

Comme dans l'algorithme de Lumer et Faieta, les fourmis évoluent sur une grille G, contenant des objets dispersés aléatoirement. Les points introduits pour l'amélioration de l'algorithme L.F peuvent être résumés comme suit :

- La grille sur laquelle évoluent les fourmis artificielles est toroïdale, c'est-à-dire que les fourmis passent d'un coté à un autre en seul pas ;
- La grille de forme carrée, est de dimension égale à $L \times L$, avec $L = \sqrt{2N}$ et N représente le nombre d'objets présent sur la grille ;
- Une case de la grille, peut contenir plusieurs objets, contrairement à l'algorithme LF ;
- Une fourmi peut transporter plusieurs objets à la fois ;
- Une fourmi a_i, possède une patience $p(a_i)$ lui permettant dans le cas où tous les objets de la grille sont transportés par des fourmis, de déposer le(s) objet(s) qu'elle transporte ;
- Chaque fourmi, possède une mémoire gérée en FIFO, qui lui permet une comparaison entre les centres de gravité des objets mémorisés (préalablement transportés) et celui des objets transportés, puisque une fourmi peut ramasser plusieurs objets. Cela lui permet de se diriger vers la case des objets les plus similaires.

L'algorithme *AntClass* peut être représenté comme suit :

Algorithme 5 Algorithme AntClass [Monmarché, 2000].

pour $T = 1$ à T_{max} **faire**
  pour $k = 1$ à A **faire**
    Déplacer la fourmi a_k sur une case non occupée par une autre fourmi
    si il y a un tas d'objets T_j sur la même case que a_k **alors**
      si la fourmi transporte un objet O_i (ou un tas d'objets T_i) **alors**
        Déposer l'objet O_i (ou le tas T_i) transporté par la fourmi sur le tas T_i suivant la probabilité $p_d(O_i, T_j)$ (ou $p_d(O_i, T_j)$)
      sinon
        //La fourmi ne transporte pas d'objets
        Ramasser l'objet O_i le plus dissimilaire du tas T_i (jusqu'à ce que la capacité $O(a_k)$ de la fourmi soit atteinte ou que le tas soit vide) selon la probabilité $P_p(T_j)$.
      finsi
    finsi
  fin pour
fin pour

4.5.3.3 Algorithme AntTree

L'algorithme AntTree est inspiré du principe d'auto assemblage observé dans une colonie de fourmis. Dans cet algorithme, les fourmis commencent à se fixer sur un support initial, puis les autres fourmis qui arrivent se fixent les unes sur les autres selon des indices de similarité caractérisant les fourmis.

En finalité ce regroupement représente la classification des objets. Cet algorithme est présenté avec plus de détails dans [Azzag et al., 2003].

4.5.3.4 Algorithme AntClust

C'est un algorithme inspiré de la reconnaissance par odeur des fourmis appartenant à la même colonie. Dans cet algorithme, une fourmi est associée à une donnée à classer, ainsi, la fourmi tente de retrouver ses « proches » (par conséquent les données appartenant à la même classe) dans un processus de rencontres aléatoires avec d'autres fourmis. Pour une description détaillée de cet algorithme voir [Labroche, 2003].

4.6 Systèmes immunitaires artificiels pour la classification de données

Dans cette section, nous allons nous intéresser a un domaine particulier de l'optimisation : la classification des données multidimensionnelles. Nous présentons une méthode biomimétique inspirée des systèmes immunitaires des vertébrés.

Dans un premier temps, nous explorons des aspects de base du système immunitaire des vertébrés et passons en revue un modèle de système immunitaire artificiel avec pour but principal de grouper et de filtrer des données brutes décrites par des échantillons multidimensionnels. L'objectif n'est pas de reproduire avec exactitude des phénomènes immunitaires réels, mais de montrer comment des concepts inspirés du système immunitaire peuvent être employés pour développer des outils informatiques pour la classification de données.

Par la suite, nous présentons un algorithme inspiré de ce modèle : le réseau immunitaire atitificiel (AiNet[1]), qui est capable de réduire la redondance et de décrire la distribution des données. Cet algorithme sera par la suite adapté pour résoudre le problème principal de ce mémoire : *la segmentation d'images*.

[1] Artificial Immune Network

4.6.1 Fondements d'un modèle de réseau immunitaire artificiel

Le système immunitaire des vertébrés a inspiré plusieurs théories relatives au traitement de l'information. Parmi ces derniers, nous pouvons citer :

La théorie des réseaux immunitaires : Qui présume des activités des cellules immunitaires, de l'apparition de la mémoire, et de la discrimination entre nos propres cellules (connues comme le soi, ou « self ») et les envahisseurs externes (connus comme non-soi « non-self »). Elle suggère également que le système immunitaire dispose d'une image interne de tous les microbes et pathogènes existants (nonself infectieux) auxquels il pourrait être exposé.

La sélection par clonage : Le principe de sélection par clonage propose une description de la manière avec laquelle le système immunitaire fait face aux microbes et pathogènes et présente une réaction immunitaire adaptative.

Le principe de la maturation d'affinité : Le principe de maturation d'affinité explique comment le système immunitaire devient de plus en plus apte à identifier et à éliminer les microbes et pathogènes (substances antigéniques).

Dans cette section, nous décrirons ces théories et montrerons que plusieurs de leurs concepts peuvent être employés pour développer un modèle de réseau immunitaire artificielle, appelé AiNet.

Le modèle AiNet se compose d'un ensemble de cellules, appelées « anticorps », reliées entre-elles par des liens pondérés. Les anticorps sont censés représenter l'image interne des microbes et pathogènes (données d'entrée) contenus dans l'environnement auquel le système est exposé. Les liens entre les anticorps déterminent leurs interdépendances, fournissant ainsi un degré de similitude (dans un espace métrique donné) : plus les anticorps sont liés, plus ils sont semblables.

Basé sur un ensemble de motifs non étiquetés, où chaque motif (objet, ou échantillon, ou donnée en entrée) est décrit par des variables (des attributs ou des caractéristiques), un réseau de cellules immunitaires sera construit pour répondre à des questions telles que :

1. Y a-t-il une grande quantité de redondance dans les données et, s'il y en a, comment pouvons-nous la réduire ?
2. Y a-t-il un groupe ou un sous-groupe intrinsèque aux données ?
3. Combien de groupes y a-t-il dans l'échantillon ?
4. Quelle est la structure ou la distribution spatiale de ces données (groupes) ?

5. Comment pouvons-nous produire des règles de décision pour classifier des échantillons de données ?

4.6.2 Principes des systèmes immunitaires

Dans un premier temps, nous esquisserons quelques aspects du système immunitaire adaptatif humain. Un certain nombre de concepts et de limites techniques seront présentés pour introduire la terminologie. Les détails de la théorie des réseaux immunitaires, de la sélection par clonage et du principe de maturation d'affinité seront donnés dans les sections consacrées. Le lecteur se référera à [Janeway et al., 2000] qui est un bon texte d'introduction en immunologie et [Castro and Zuben, 1999] pour l'immunologie sous la perspective de l'intelligence artificielle.

Le système immunitaire est un ensemble de cellules, de molécules et d'organes ayant comme rôle de limiter les dommages causés par les microbes et pathogènes, appelés *antigènes* (Ag), qui provoquent une réaction immunitaire. Un type de réponse est la sécrétion de molécules d'*anticorps* (Ab) par des cellules dites « *de type B* », ou des *lymphocytes « de type B »*. Les anticorps sont des molécules réceptrices en «Y», accrochées à la surface d'une cellule de type B dont le rôle principal est la reconnaissance, via une correspondance complémentaire, de l'antigène. Les molécules d'anticorps identifient une partie de l'antigène appelé *l'épitope*. Les anticorps présentent également des épitopes, qui sont appelés des *idiotopes*. Un ensemble d'idiotopes constitue un *idiotype*. Tandis que chaque cellule de type-B ne peut avoir qu'un seul type d'anticorps, les antigènes ont typiquement plusieurs types d'épitopes, et peuvent donc être reconnus par plusieurs anticorps.

La partie de l'anticorps responsable de la détection (reconnaissance) d'antigène s'appelle le *paratope*, également connu sous le nom de région V (pour région variable). Une région V est variable parce qu'elle peut se déformer pour améliorer la correspondance (complémentarité) avec un antigène donné.

La force et la spécificité de l'interaction *Ag-Ab* sont mesurées par l'affinité (complémentarité de niveau) de leur correspondance. Voir la figure 4.13 pour une illustration d'un antigène avec ses nombreux épitopes et d'un anticorps avec son paratope et son idiotope.

Une fois stimulée, la cellule de type B prolifère et sécrète ses molécules réceptrices en tant qu'anticorps libres. Les anticorps peuvent être ainsi des molécules libres ou des récepteurs attachés aux cellules. La sécrétion exige que les cellules $type - B$ deviennent actives, subissent la prolifération (clonage) puis se séparent finalement en des *cellules de plasma* et des *cellules mémoire*.

Un *clone* est une cellule, ou un ensemble de cellules, qui est la progéniture de la

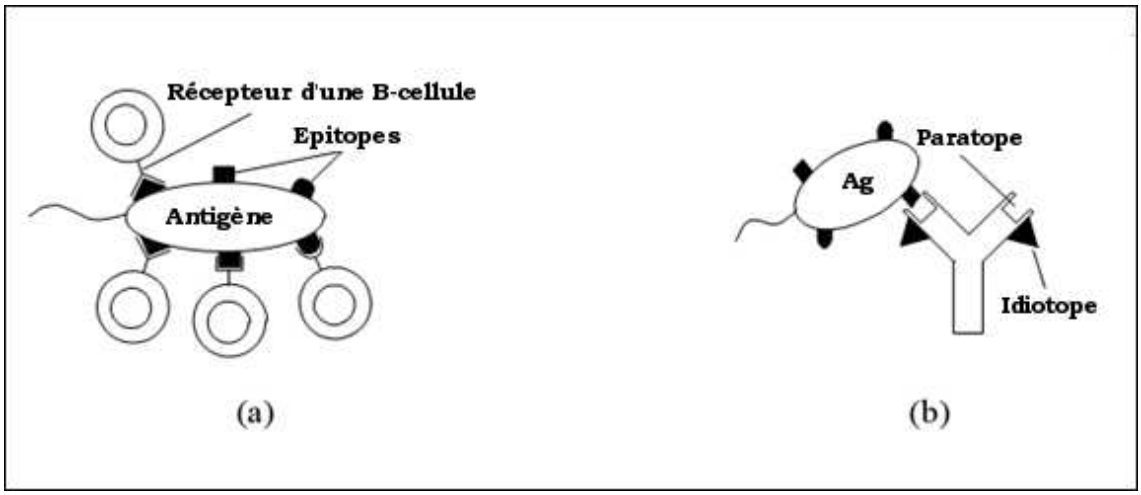

FIG. 4.13 – Cellule de type B, antigène, anticorps, épitopes, paratopes et idiotopes. (a) Un antigène avec ses épitopes multiples identifiés par différentes cellules de B. (b) Anticorps portant un paratope (ou région V), et son idiotope.

même cellule mère.

Une *cellule de plasma* est une cellule capable de sécréter des anticorps dont le taux de correspondance est élevé. Une *cellule mémoire* est une cellule présentant une affinité élevée avec l'antigène. Elle sera sauvegardée pour garantir une réponse plus rapide et plus forte face à un antigène précédemment rencontré. Ces cellules précieuses pour le système, se développent en concentration et en affinité (maturation d'affinité), alors que celles n'ayant pas déclancher une réaction immunitaire s'éteignent.

Ce processus de base de reconnaissance de forme et de choix est connu sous le nom de la *sélection par clonage* [Burnet, 1978]. Pour être efficace, le système immunitaire doit apprendre à distinguer entre les cellules du système immunitaire et les corps étrangers. Ce processus s'appelle discrimination soi/non-soi. Les cellules identifiées comme « cellules-hôtes » ne déclenchent pas de réaction immunitaire, le système leur est tolérant, alors que celles qui ne le sont pas provoquent une réaction ayant pour résultat leur élimination.

4.6.3 Théorie des réseaux immunitaires

La théorie des réseaux immunitaires, comme proposée par [Jerne, 1974], fait état d'un point de vue nouveau sur les activités lymphocytes, de production d'anticorps, de choix de répertoire immunitaire, de tolérance et de discrimination self/nonself, de mémoire et de l'évolution du système immunitaire.

La théorie des réseaux immunitaires suggère que le système immunitaire se com-

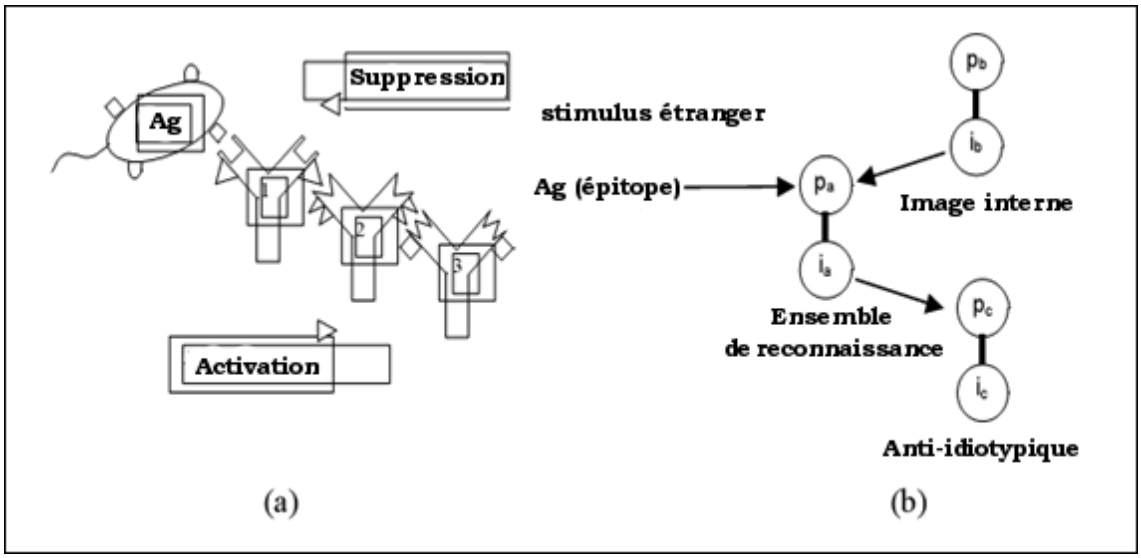

FIG. 4.14 – Représentations de réseaux d'Idiotypes. (a) Un antigène stimule la production d'anticorps de la classe a, qui stimule la classe b, et ainsi de suite. (b) Représentation d'une vue détaillée de réseau d'idiotypes.

pose d'un réseau de cellules capables de s'identifier entre elles et ainsi de réagir même en l'absence d'antigènes.

Le système immunitaire est ainsi perçu comme un réseau complexe de paratopes qui identifient des ensembles d'idiotopes, eux-mêmes identifiés par des ensembles de paratopes. Ainsi un paratope peut reconnaître et être reconnu.

Les événements appropriés dans le système immunitaire sont non seulement les molécules, mais également leurs interactions. Les cellules immunitaires peuvent répondre positivement ou négativement au signal d'identification d'un antigène ou de toute autre cellule immunitaire ou molécule.

Une réponse positive entraîne la prolifération de cellules, leur activation et la sécrétion d'anticorps, alors qu'une réponse négative mène à la tolérance de la cellule reconnue et la suppression de la cellule immunitaire (voir le schéma 4.14(a)).

La théorie des réseaux immunitaires peut être résumée comme suit (voir le schéma 4.14(b)).

Quand le système immunitaire est exposé à l'antigène (Ag), son épitope est identifié (avec divers degrés de correspondance) par un ensemble de paratopes, appelé P_a.

L'ensemble des idiotopes l_b est appellé image interne de l'épitope (ou de l'antigène) parce qu'il est identifié par le même ensemble P_a qui a identifié l'antigène.

L'ensemble l_b est associé à un ensemble P_b des paratopes se trouvant sur les molécules et sur les récepteurs de cellules. En outre, chaque idiotope de l'ensemble

l_a est identifié par un ensemble de paratopes, de sorte que l'ensemble l_a tout entier soit lui-même identifié par un ensemble encore plus grand P_c de paratopes.

En suivant ce schéma, nous obtenons des ensembles toujours plus grands qui reconnaissent ou sont reconnus par les ensembles précédemment définis dans le réseau. Les flèches (voir le schéma 4.14(b)) indiquent un effet « stimulateur » quand des idiotopes sont identifiés par des paratopes sur des récepteurs de cellules et un effet « suppressif » quand les paratopes identifient des idiotopes sur des récepteurs de cellules.

4.6.4 Sélection par clonage et maturation d'affinité

L'apprentissage dans le système immunitaire impose d'augmenter la taille de la population des cellules immunitaires et l'affinité des lymphocytes qui se sont avérés intéressants pour avoir identifié un antigène.

Durant la vie d'un individu, son système immunitaire peut être exposé à un antigène donné à plusieurs reprises. L'exposition initiale à un antigène stimule une réaction immunitaire adaptative qui est assurée par un « petit » éventail de copie de cellules de type B, chacune produisant un anticorps dont l'affinité est différente. L'efficacité de la réaction immunitaire à l'exposition suivante est considérablement améliorée par le stockage d'un certain nombre d'anticorps présentant une bonne correspondance provenant de la première infection (cellules mémoire), et ce afin de former un ensemble initial de clone d'affinité élevée pour une éventuelle exposition ultérieure : [Ada and Nossal, 1987].

C'est un schéma intrinsèque d'une stratégie d'étude de renfort, où le système améliore de façon continue sa faculté à accomplir sa tâche (identifier les antigènes) [Sutton and Barto, 1998].

Les anticorps participant à une réponse de mémoire ont des affinités plus élevées que celles de la première réponse. Ce phénomène est désigné sous le nom de la maturation de la réaction immunitaire. Cette maturation exige que les emplacements liants les molécules d'anticorps soient structurellement différents de ceux de la réponse primaire.

Des changements aléatoires (mutations) se produisent dans la région variable et entraînent éventuellement une augmentation de l'affinité de l'anticorps. Ceux sont ces cellules de forte affinité qui sont choisies pour faire partie du répertoire des cellules de mémoire.

Les cellules présentant une affinité insuffisante, ou provoquant une réaction immunitaire, doivent être éliminées. Au lieu de la suppression par clonage prévue pour toutes les cellules réagissant aux cellules hôtes, les lymphocytes de type B peuvent

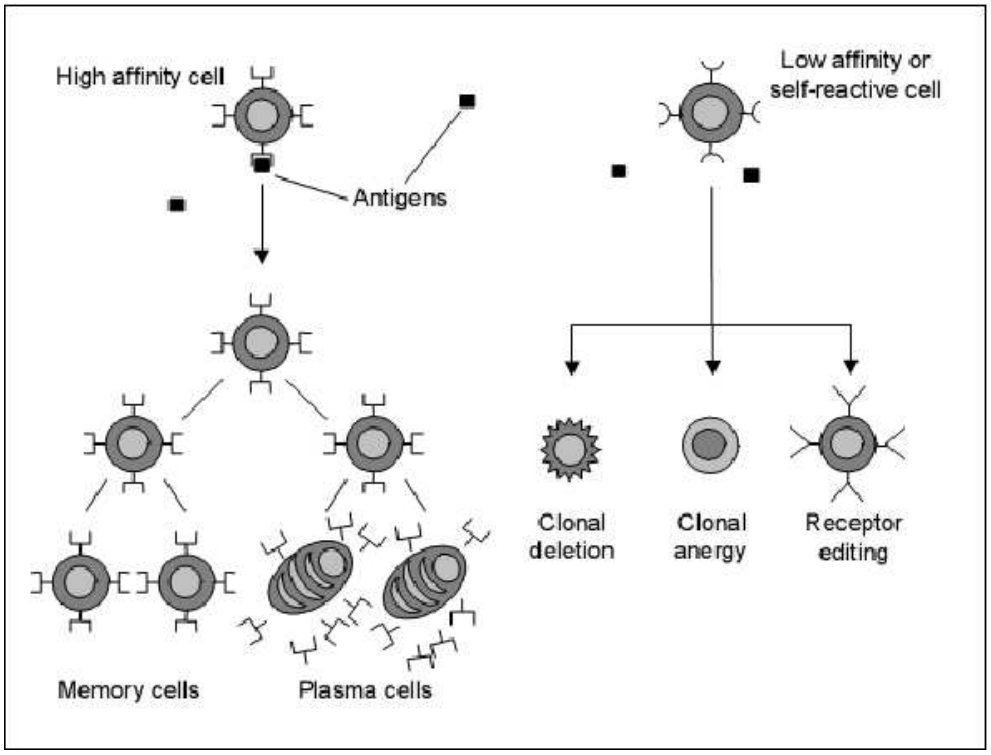

FIG. 4.15 – Interactions antigéniques avec des lymphocytes [Castro and Zuben, 2001].

subir l'édition de leurs récepteurs : les type B remplacent leurs récepteurs self-reactifs et produisent des récepteurs entièrement nouveaux. Certains des résultats possibles d'une exposition d'un lymphocyte à un antigène sont résumés dans la figure 4.15.

Une accumulation rapide des mutations (hyper-mutation) est nécessaire pour une maturation rapide de la réaction immunitaire, mais la majorité des changements mènent à des anticorps moins efficaces ou non fonctionnels. Si une cellule qui vient juste de subir une mutation utile continue à muter au même taux, alors l'accumulation des changements peut causer la perte de la mutation avantageuse.

Le mécanisme de sélection par clonage peut fournir les moyens par lesquels le contrôle du processus d'hyper-mutation est rendu dépendant de l'affinité du récepteur. Les cellules d'affinité moindre, ont plus de chance d'être mutées et éliminées si leur affinité aux antigènes demeure faible.

Cependant, en ce qui concerne les cellules dont les récepteurs présentent un forte affinité, la mutation peut être graduellement désactivée [Kepler and Perelson, 1993].

4.6.5 AiNet : Un modèle de réseau immunitaire artificiel pour la classification de données

Afin de mesurer la correspondance immunitaire, nous considérons tous les événements immunitaires comme se produisant dans un espace, qui est un espace métrique multidimensionnel où chaque axe représente une mesure physico-chimique caractérisant une molécule [Perelsen and Oster, 1979].

Nous supposons un ensemble de mesures, dépendant du problème, pour caractériser une configuration moléculaire comme un point s dans S. Par conséquent, un point dans un espace L-dimensionnel, appelé espace de forme, indique l'ensemble de mesures nécessaires pour déterminer les interactions d'anticorps-anticorps ($Ab-Ab$) et d'antigène-anticorps ($Ag-Ab$).

Mathématiquement, cette structure (établissement de mesures qui définissent un anticorps ou un antigène) peut être représentée comme une chaîne (ou un vecteur) L-dimensionnelle. Les interactions possibles dans AiNet seront représentées sous forme de graphique de connectivité. Le modèle de réseau immunitaire artificiel peut être formellement défini comme suit :

Définition 1 *AiNet est un réseau pondéré, pas nécessairement entièrement relié, composé d'un ensemble de nœuds, appelés les anticorps, et d'un ensemble de paires de nœuds appelées les connections associées à un poids (force de raccordement).*

Les clusters d'AiNet représentent l'image interne des groupes existants dans les données. Comme illustration, supposons un ensemble de données composé de trois clusters, selon la figure 4.16(a).

Une architecture de réseau hypothétique produite par l'algorithme est montrée dans la figure 4.16(b).

Les nombres dans les cellules indiquent leurs étiquettes (le nombre total est généralement plus grand que le nombre de cluster et beaucoup plus petit que la taille de l'échantillon), les nombres portés à côté des raccordements représentent leur poids, et les lignes en pointillées indiquent des raccordements à supprimer, afin de détecter les clusters et de définir la structure finale du réseau.

Nous notons la présence de trois clusters distincts, chacun comprenant un nombre différent d'anticorps et de liaisons. Ces clusters représentent l'ensemble des données originales.

Nous notons également que le nombre d'anticorps dans le réseau est beaucoup plus réduit que la taille de l'échantillon de données, caractérisant ainsi une architecture appropriée à la compression de données.

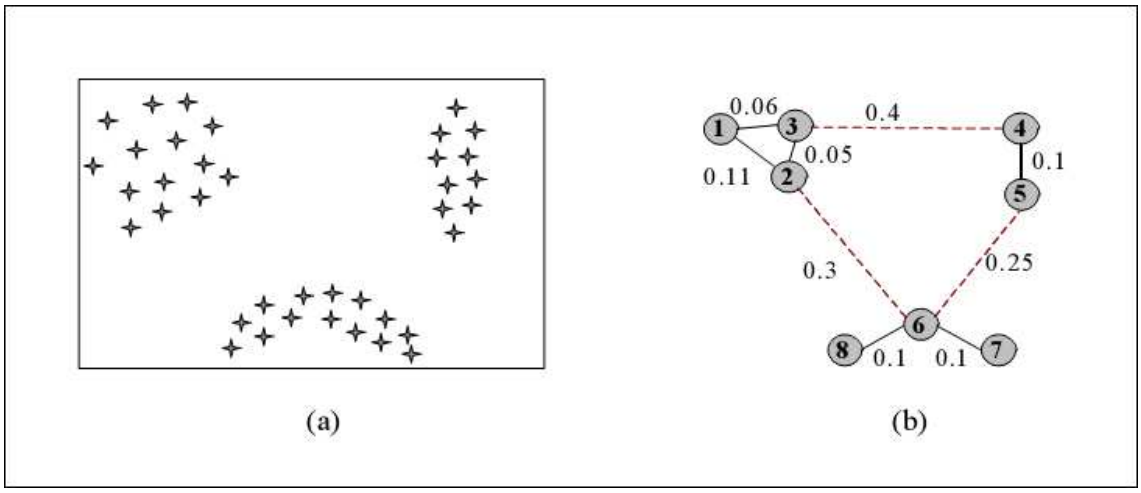

FIG. 4.16 – illustration d'ainet. (a) échantillon disponible avec trois clusters de densité élevée. (b) réseau des cellules marquées avec leurs forces de raccordement assignées aux liens. les lignes en pointillées indiquent des raccordements à supprimer afin de produire des sous-graphes indépendants.

En conclusion, la forme de la distribution spatiale des anticorps est similaire à celle de la distribution spatiale antigénique.

De même que dans les modèles de [Castro and Zuben, 2001], nous ne faisons aucune distinction entre les cellules du réseau et leurs molécules extérieures (anticorps).

Les interactions *Ag-Ab* et *Ab-Ab* sont évaluées par des mesures de proximité (ou similitude), le but étant d'employer une distance métrique pour produire un répertoire d'anticorps qui constitue l'image interne des antigènes à reconnaître et d'évaluer le degré de similitude parmi les anticorps d'Ainet de façon que la cardinalité du répertoire puisse être contrôlée. Ainsi, l'affinité *Ag-Ab* est inversement proportionnelle à la distance qui les sépare : plus la distance est petite, plus l'affinité est grande, et réciproquement.

Il est important de préciser que, dans le système immunitaire biologique, l'identification se produit par une correspondance complémentaire entre un antigène donné et un anticorps. Néanmoins, dans beaucoup d'applications de systèmes immunitaires artificiels, y compris dans ce modèle, la génération d'un répertoire d'anticorps basée sur la similarité (au lieu de la complémentarité) est un choix opportun.

Comme proposé dans la théorie originale des réseaux immunitaires, les cellules existantes se concurrencent pour l'identification antigénique, celles qui réussissent entraînent la prolifération et l'activation de cellules du réseau (selon le principe de sélection par clonage décrit en section 4.6.2), alors que celles qui échouent sont éliminées.

En outre, l'identification $Ab - Ab$ aura comme conséquence la suppression de

réseau. Dans notre modèle, la suppression est effectuée en éliminant les anticorps self-réactifs, étant donné un seuil de suppression σ_s. Chaque paire (Ag_j, Ab_i) $i = 1, \ldots, N$ $j = 1, \ldots, M$, sera en relation dans l'espace S via l'affinité $d_{i,j}$ qui les relies, et qui représente la probabilité de déclencher une réponse immunitaire. De même, une valeur d'affinité $s_{i,j}$ sera assignée à chaque pair $Ab_j - Ab_i$ $i, j = 1, \ldots, N$, reflétant leurs interactions (similitude).

La notation suivante est adoptée :

Ab : répertoire disponible d'anticorps $Ab \in S^{N \times L}$ $Ab = Ab\{d\} \bigcup Ab = Ab\{m\}$;

$Ab\{m\}$: répertoire total d'anticorps mémoire $(Ab\{m\} \in S^{N \times m}$ $m \leq N)$;

$Ab\{d\}$: nouveaux anticorps à insérer dans Ab $(Ab\{d\} \in S^{N \times L})$;

Ag : population des antigènes $(Ag \in S^{m \times L})$;

f_j : vecteur contenant l'affinité de tous anticorps Ab_i $(i = 1, \ldots, N)$ avec l'antigène Ag_j. L'affinité est inversement proportionnelle à la distance $Ag - Ab$;

S : matrice de similitude entre chaque ciseaux $Ab_i - Ab_j$, dont les éléments sont $s_{i,j}$ $(i, j = 1, \ldots, N)$;

C : la population de clone génères a partir de Ab, $(C \in S^{N \times L})$;

$C*$: population C après le procédé de maturation d'affinité ;

d_j : vecteur contenant l'affinité entre chaque élément de l'ensemble $C*$ avec Ag_j ;

ζ : pourcentage des anticorps mûrs à choisir ;

M_j : clone de mémoire pour l'antigène Ag_j (restant du processus de la suppression clonale) ;

M_j* : mémoire clonale résultante pour l'antigène Ag_j ;

σ_d : seuil de la mort naturelle ;

σ_s : seuil de suppression.

L'algorithme AiNet vise à établir une mémoire qui reconnaît et représente l'organisation structurale des données en entrée. Plus les anticorps sont dispersés, moins parcimonieux sera le réseau (bas taux de compression), tandis que plus les anticorps sont regroupés, plus le réseau sera réduit (taux de compression amélioré).

Le seuil de suppression (σ_s) commande le niveau de spécificité de l'anticorps, l'exactitude de la classification, et la topologie du réseau.

Voir L'algorithme 6 pour une description étapes par étapes d'AiNet.

Algorithme 6 Algorithme Ainet

pour $T = 1$ à T_{max} **faire**

pour tout Motif antigenique Ag_i **faire**

(a) Déterminer son affinité $f_{i,j}$ $i = 1, \ldots, N$ pour chaque anticorps Ab_j du répertoire $f_{i,j} = 1/D_{i,j}$ $i = 1, \ldots, N$, $D_{i,j} = \|Ab_i - Ag_i\|$ $i = 1, \ldots, N$

(b) Un sous-ensemble $Ab\{n\}$ composé des n anticorps dont l'affinité est la plus élevée est choisie

(c) Les n anticorps choisis vont proliférer (se cloner) proportionnellement à leur affinité $f_{i,j}$, produisant un ensemble C de clones : plus l'affinité est importante, plus le nombre de clones pour chaque anticorps est grand.

(d) L'ensemble C est soumi à un procédé dirigé de maturation d'affinité (mutation guidée) produisant d'un ensemble muté $C*$, où chaque anticorps k de $C*$ subi une mutation de taux a_k inversement proportionnel à l'affinité antigénique $f_{i,j}$ de son anticorps de parent : plus l'affinité est importante, plus le taux de mutation est faible.

(e) Déterminer $d_k = 1/D_{k,j}$ le taux d'affinité entre Ag_j et tous éléments de $C*$: $D_{k,j} = \|Ag_j - C_k*\|$

(f) A partir de $C*$, re-sélectionner $\zeta\%$ des anticorps dont le facteur d_k est le plus élevé, et les mettre dans une matrice M_j de mémoire clonale

(g) Eliminez tous les clones de M_j dont l'affinité $D_{k,j} < \sigma_d$

(h) Déterminez l'affinité $s_{i,k}$ entre les clones mémoire : $s_{i,k} = \|M_{i,j} - M_{k,j}\|$

(i) Eliminez de la mémoire les clones dont le parametre $s_{i,k} < \sigma_s$

(j) Combiner la matrice totale de pixels avec les clones M_j* pour Ag_j : $Ab\{m\} = [Ab\{m\}, M_j*]$

fin pour

Déterminez l'affinité entre tous les pixel de mémoire d'Abm : $s_{i,k} = \|Ab\{m\}_i - Ab\{m\}_j\|$ pour chaque i et j

(Suppression de réseau) Garder un seul exemplaire des couples de pixel pour lesquelles $s_{i,k} < \sigma_s$

Construire la matrice totale d'anticorps $Ab = [Ab\{m\}, Ab\{d\}]$

fin pour

4.7 Conclusion

Dans ce chapitre, quelques méthodes d'optimisations ont été présentées. Ces méthodes peuvent être appliquées à divers domaines tels les bases de données, l'imagerie, ou la recherche opérationnelle. Toutes les méthodes présentées sont des méthodes d'optimisation inspirées du vivant, elles sont génériques et peuvent être adaptées à divers types de problèmes, après avoir définit les différents paramètres nécessaires à l'algorithme à appliquer en fonction du problème à traiter.

Chapitre 5

Conception et mise en œuvre

5.1 Introduction

Nous présentons dans ce chapitre la conception de notre application. La première section consiste en une introduction à la plateforme de développement utilisée dans le cadre de notre projet : GIMP. Les sections suivantes porteront sur l'architecture générale de notre application, ainsi que sur la description détaillée des différents algorithmes de segmentation implémentés.

5.2 Présentation de GIMP

GIMP est un outil portable de manipulation d'images. « GIMP » est un acronyme de « GNU Image Manipulation Program » (Programme GNU de Manipulation d'Images).

GIMP est utilisable pour une grande variété de tâches de manipulation d'images comme la retouche photographique, la composition ou la création d'images.

Il offre de nombreuses fonctionnalités. Il peut être utilisé comme :

- un simple programme de dessin ;
- un programme de retouche photo ;
- un système en ligne de traitement par lot ;
- un générateur d'images pour la production en masse ;
- Un convertisseur d'images d'un format vers un autre, etc.

GIMP est extensible. Il est possible d'étendre ses fonctionnalités en installant des « Greffons » (plug-ins), qui sont de petits programmes externes qui viennent se greffer sur l'application principale pour lui ajouter de nouvelles fonctionnalités.

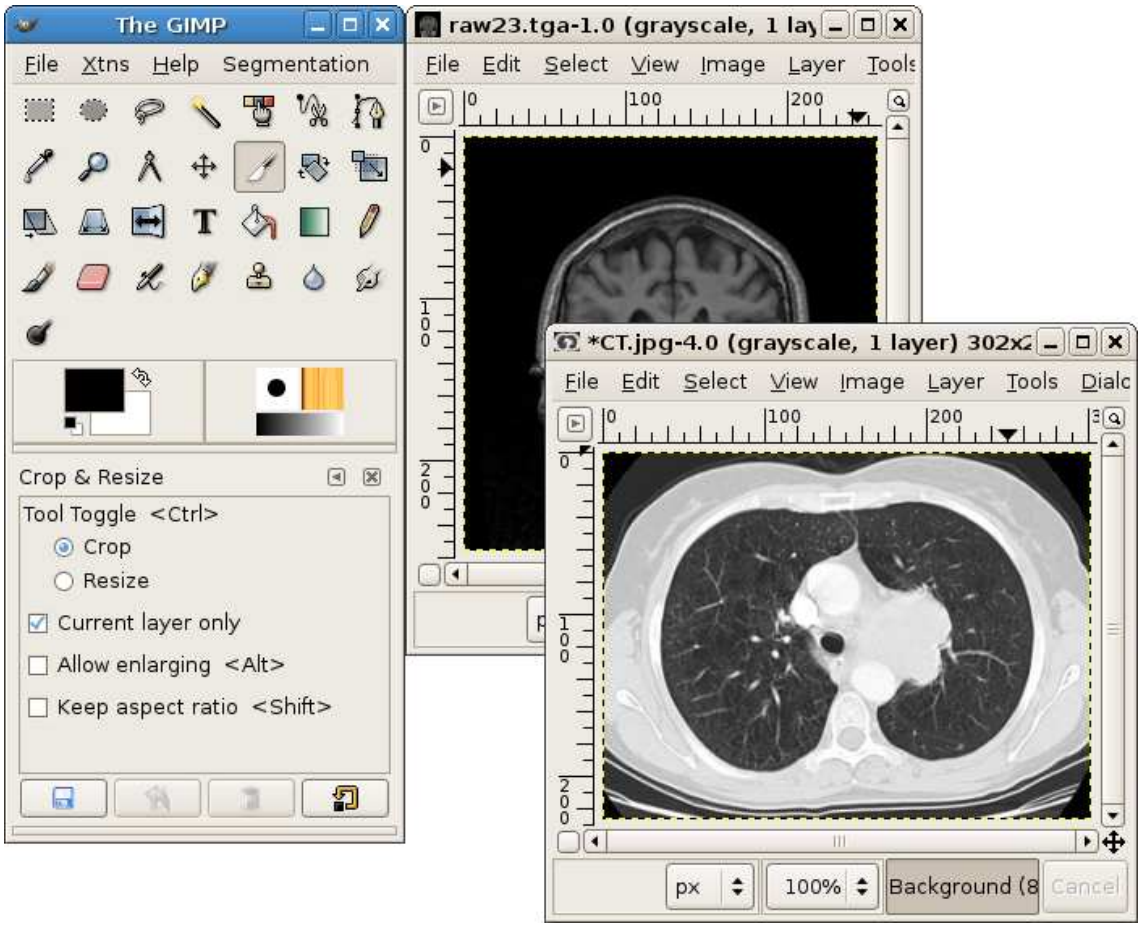

FIG. 5.1 – Boite à outils de GIMP

Une des forces de GIMP est sa libre[1] disponibilité depuis de nombreuses sources pour de nombreux systèmes d'exploitation. La plupart des distributions GNU/Linux incluent GIMP en standard. GIMP est aussi disponible pour d'autres systèmes d'exploitation comme Microsoft Windows ou Mac OS X d'Apple (Darwin). Une caractéristique importante de Gimp est qu'il n'est pas un logiciel propriétaire. C'est un logiciel libre couvert par la licence GNU GPL[2]. Cette licence offre à l'utilisateur (entre autres) la possibilité de lire et de modifier le code source qui compose le programme.

La figure 5.1 montre la boite à outils du programme GIMP.

[1]Le terme libre ici fait réference à la liberté non pas à la gratuité

[2]GNU General Public Licence, qui est une licence libre écrite par Richard Stallman dans le cadre du «GNU project», elle est dans sa troisième version lancée le 29 juin 2007 (voir http ://www.gnu.org)

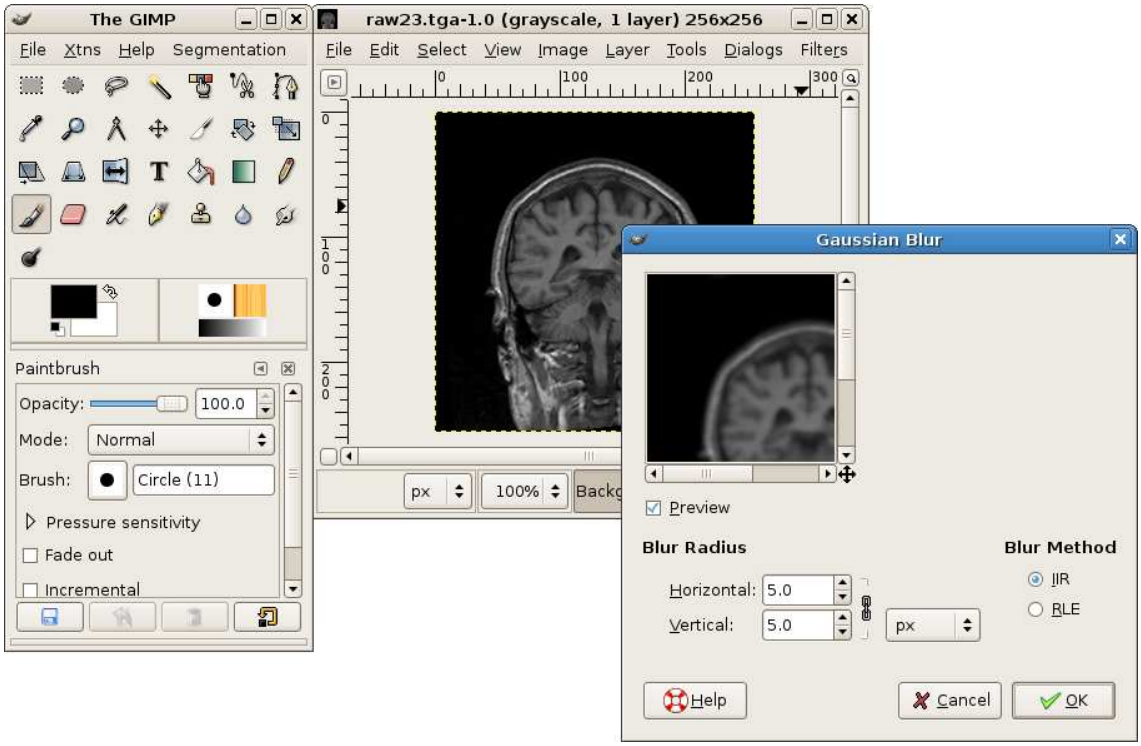

FIG. 5.2 – Greffon « flou gaussien » dans le logiciel GIMP.

5.3 Système de greffons « plug-ins »

Dans cette section, nous allons présenter une fonctionnalité importante du logiciel GIMP qui est le système de greffons. Cette fonctionnalité nous a permis d'intégrer dans GIMP notre plateforme Son principal avantage est qu'il est beaucoup plus simple d'ajouter une fonctionnalité en écrivant un greffon plutôt qu'en modifiant les lignes de codes qui constituent le noyau de GIMP. de segmentation d'images.

Les greffons sont des programmes externes qui tournent sous le contrôle de GIMP et qui sont en étroite interaction avec lui. Les greffons peuvent manipuler une image de la même manière que l'utilisateur.

Dans la distribution principale de GIMP, plusieurs greffons sont installés par défaut. La plupart sont accessibles via le menu Filtres, mais bon nombre se répartissent dans d'autres menus. A titre d'exemple, un des greffons présents dans le menu Filtres de GIMP est la fonction « Flou gaussien... », qui rend une image floue en calculant une moyenne des niveaux en chaque point, pondérée par une gaussienne.

La figure 5.2 montre un exemple d'application du greffon « Flou gaussien... ».

5.4 Architecture générale

Dans le cadre de notre projet, nous nous proposons de développer un greffon (plugin) pour GIMP, baptisé «*Multipass*», permettant de tester et de comparer plusieurs algorithmes de segmentation suivant des critères tels que la vitesse d'exécution, la qualité des résultats, la sensibilité au bruit, etc. GIMP fournira l'image en entrée à « *Multipass* » qui lui rendra résultat après application de la segmentation (voir figure 5.3).

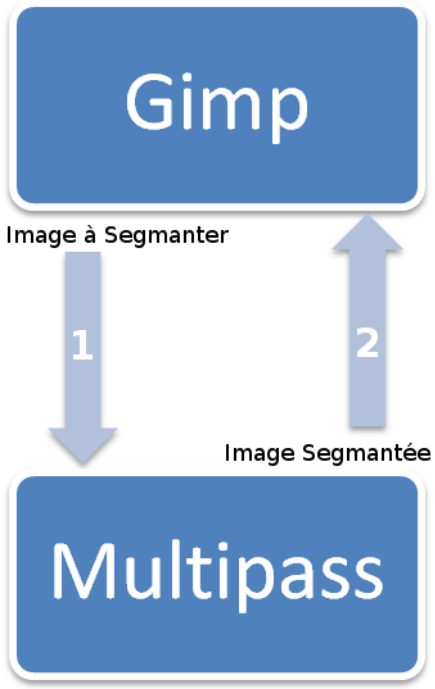

FIG. 5.3 – Communication entre Multipass et GIMP

En utilisant GIMP comme plateforme de développement, nous profitons d'un logiciel d'édition d'images, riche, complet et libre de droit, garantissant un résultat final offrant toutes les fonctionnalités de base d'édition d'images à savoir :

- ouverture et lecture des images d'une vingtaine de formats différents ;
- conversion inter-format ;
- édition et synthèse ;
- retouche, etc.

Cette approche permet ainsi, de se focaliser sur ce qui fait l'objet de ce mémoire : *les algorithmes de segmentation d'images.*

Spécifiquement, les algorithmes considérés seront du type classification de pixels. En effet, la plupart des algorithmes *biomimétiques*[3] sont en fait des métaheuristiques

[3]On appel biomimétisme toute démarche consistant à reproduire artificiellement des propriétés

généralement applicables à tout problème de classification ou d'optimisation combinatoire. Il est donc naturel que leur application à la segmentation conduise à les considérer comme des algorithmes de classification de pixels.

Dans le cadre de notre projet nous avons opté pour le traitement de la segmentation d'images en niveaux de gris et ce pour plusieurs raisons dont les principales sont les suivantes :

- La restriction du champ d'application ouvre en même temps la voie à plusieurs optimisations et possibilités d'extension de l'application comme nous le présenterons par la suite ;
- Notre cadre d'application étant principalement les images médicales, et ces dernières étant essentiellement en niveaux de gris, nous nous devions d'exploiter cette caractéristique pour créer une solution optimisée pour le problème, plutôt que de développer des solutions générales qui ne s'adapteraient que moyennent au problème spécifique des images en niveaux de gris.

L'architecture de *Multipass* suit un modèle en « couches » (voir la figure 5.4), comportant trois niveaux :

1. **Niveau 1 :** se compose d'un seul module :

   Interface Utilisateur : Ce module est chargé du dialogue avec l'utilisateur. Sont rôle est de construire les boites de dialogues permettant à l'utilisateur de choisir les algorithmes de segmentation ainsi que de régler leurs paramètres.

2. **Niveau 2 :** se compose des deux modules suivants :

   Interface Algorithmes : Ce module se situe à un niveau intermédiaire entre l'interface utilisateur et les algorithmes eux mêmes. Son rôle est d'assurer la communication entre les algorithmes et les éléments d'interface utilisateur. Spécifiquement, il se charge d'énumérer les algorithmes disponibles ainsi que de fournir le nombre et le type des paramètres requis par chacun d'eux pour que le module d'interface utilisateur puisse refléter ces informations à l'écran.

   Interface Fonctions de Fitness : Ce module se situe au même niveau que le précèdent et possède un rôle similaire, à la seule différence qu'il se charge de la communication entre les fonctions de fitness implémentées et le reste de l'application.

3. **Niveau 3 :** se compose des deux modules suivants :

essentielles d'un ou de plusieurs systèmes biologiques

Algorithmes de Segmentation : Au niveau le plus bas on trouve les algorithmes de segmentation eux mêmes. Les algorithmes de segmentation ne communiquent avec le reste de l'application qu'à travers le module d'Interface des Algorithmes, ce qui permet de faciliter leur développement et ainsi l'extension de l'application.

Fonctions de Fitness : De même que les algorithmes de segmentation, les fonctions de fitness se trouvent au niveau le plus bas de l'architecture et ne communique qu'avec le module d'Interface des Fonctions de fitness.

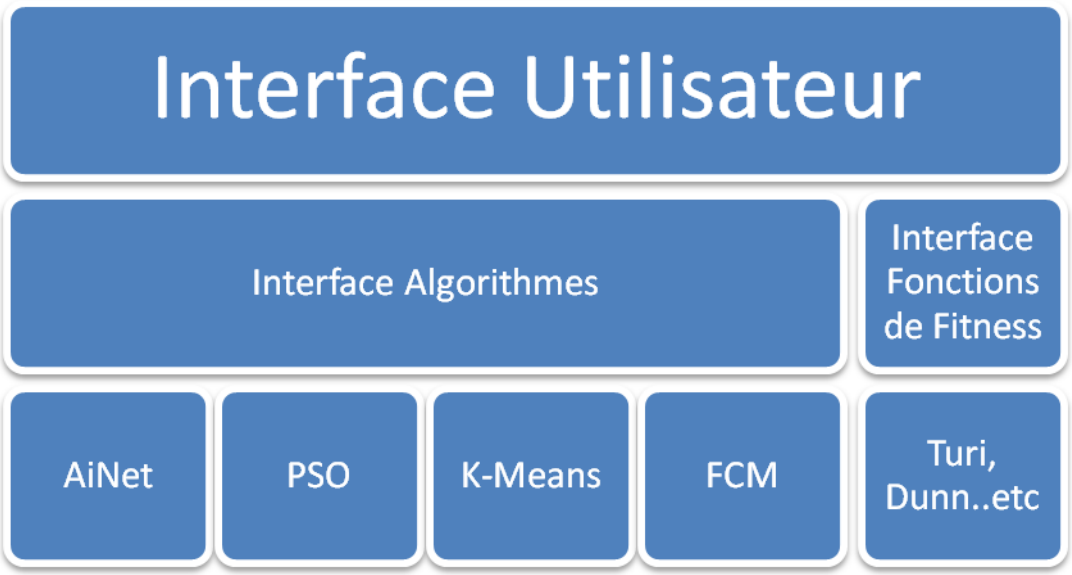

FIG. 5.4 – Architecture générale de Multipass

Comme énoncé précédemment, nous nous proposons de développer plusieurs algorithmes de segmentation afin de les comparer entre eux. Pour ce faire, une généralisation du concept « d'algorithme de segmentation » est nécessaire. La figure 5.5 présente le schéma général d'un algorithme de segmentation tel que modélisé dans le cadre de ce travail.

Comme indiqué sur la figure 5.5, un algorithme de segmentation est une « boite noire » possédant deux entrées et une sortie. En effet, tous les algorithmes développés recevront l'histogramme de l'image ainsi qu'une solution initiale et produiront une solution améliorée en se basant sur les données en entrée.

L'histogramme est une structure de données décrivant la distribution des pixels sur l'axe des niveaux de gris de l'image à segmenter. Spécifiquement, il s'agit d'un tableau de 256 entrées, chacune contenant le nombre de pixels du niveau de gris correspondant. La figure 5.6 présente un exemple d'image synthétique de 8x8 pixels.

L'image de la figure 5.6 contient : 16 pixels blancs (niveau 255), 24 pixels noirs

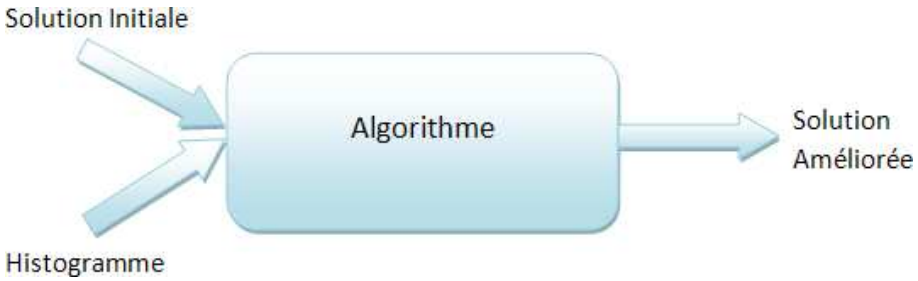

FIG. 5.5 – Schéma général d'un algorithme de segmentation.

(niveau 0) et 24 pixels gris (niveau 128). L'histogramme correspondant est donc un tableau de 256 cases toutes mises à zéro excepté les cases d'indice 0, 128 et 255 qui contiennent 24, 24 et 16 respectivement.

Comme on peut le constater, cette approche permet de condenser l'information contenue dans une image de taille quelconque, dans un tableau de 256 entiers. Ceci est rendu possible par le fait que nous nous sommes concentrés sur les images en niveaux de gris, mais aussi parce que les algorithmes étudiés sont de type classification de pixels et ne requièrent pas d'information sur la distribution des pixels sur le plan en deux dimensions de l'image.

Comme illustration, la figure 5.7 qui est apparemment différente de l'image de la figure 5.6, est caractérisée par le même histogramme que celui de la figure 5.6. Du point de vue des algorithmes étudiés ici, les images des figures 5.6 et 5.7 sont strictement identiques.

L'autre source d'information sur laquelle se baseront les algorithmes de segmentation est la solution initiale. Dans la plupart des traités de classification de données, une « solution » à un problème de classification particulier est représentée par un ensemble de centroids. Un centroid étant défini comme un point particulier sur l'espace des données à classifier et qui représente le centre d'une classe dans la solution. Sur la figure 5.8(a), les données à classifier sont représentées par des croix disposées sur un plan selon leur degré de similarité (plus les données sont similaires plus elles sont proches). La figure 5.8(b) présente un résultat possible d'application d'un algorithme de classification. Les centroids résultants sont représentés par de petits cercles noirs. Comme on peut le constater chaque donnée (croix) est associée au centroid le plus proche. De la même façon, une segmentation d'une image particulière sera modélisée comme étant l'ensemble des centroids des classes résultantes.

Dans le cas de la segmentation par classification des images en niveaux de gris, les centroids seront représentés par des niveaux de gris particuliers auxquels seront rattachés les pixels en fonction de la différence entre les niveaux de gris de ces derniers et ceux des centroids.

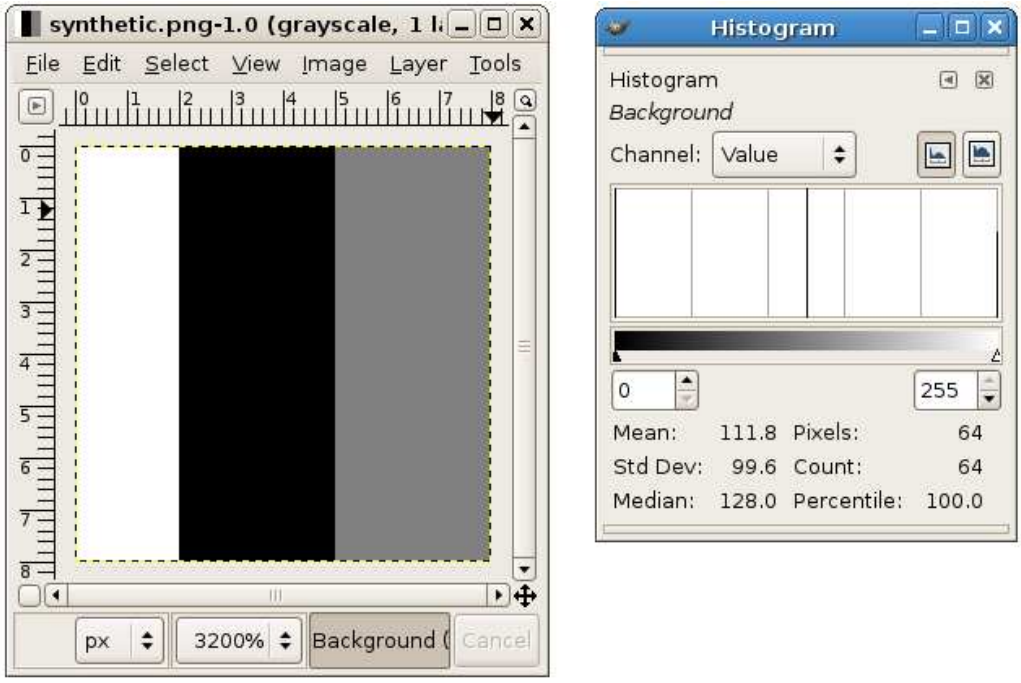

FIG. 5.6 – Image synthétique et son histogramme correspondant.

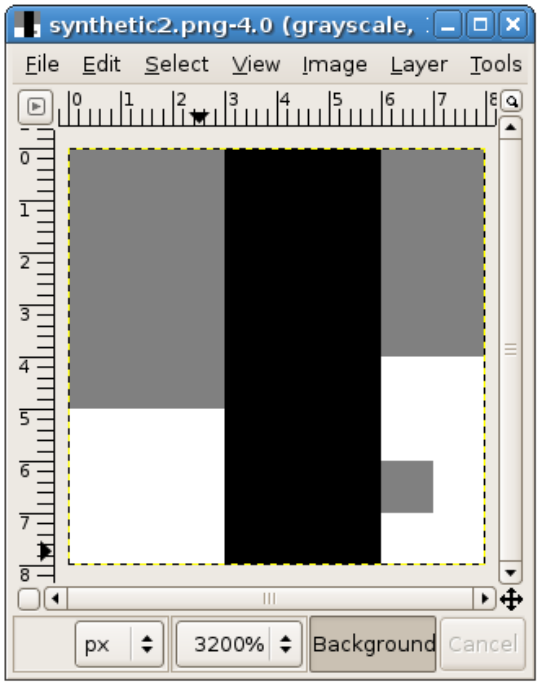

FIG. 5.7 – Image ayant le même histogramme que l'image de la figure 5.6.

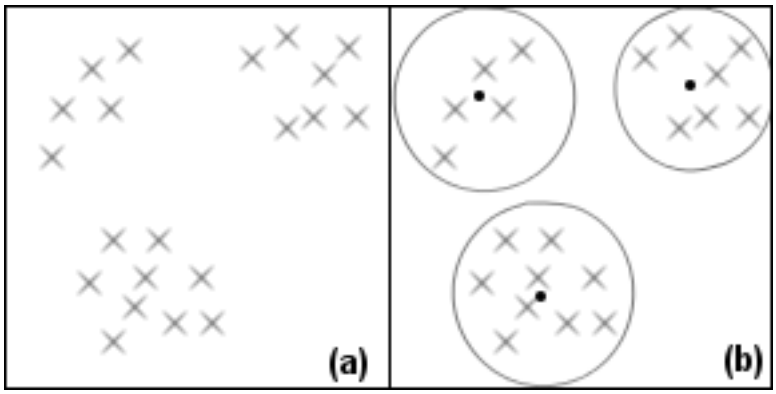

FIG. 5.8 – Classification par centroids.

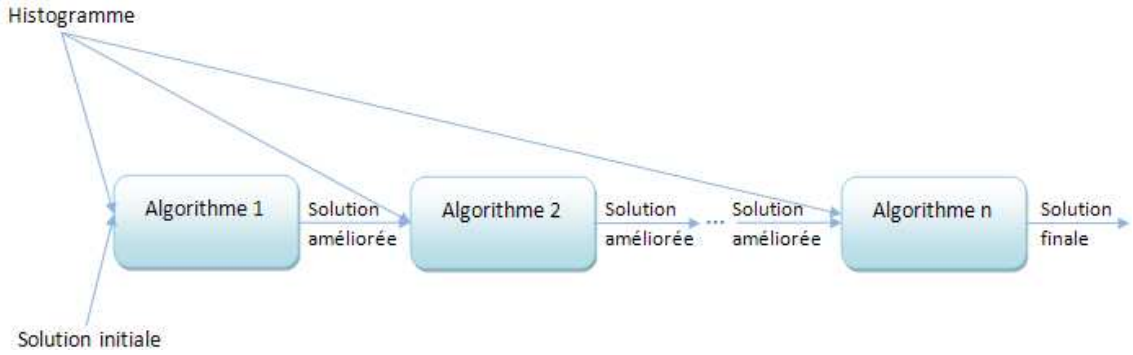

FIG. 5.9 – Hybridation des algorithmes.

L'objectif d'un algorithme de segmentation est donc de produire un ensemble de niveaux de gris représentant les centres des classes de la solution. Un autre point important, est que les algorithmes recevront également un ensemble de centroids en entrée. L'algorithme est supposé démarrer de ce point et chercher une classification optimale. Cette solution initiale, est supposée avoir été générée par un autre algorithme précédemment appliqué au même histogramme. La sortie d'un algorithme devient ainsi l'entrée d'un autre.

En effet, une des innovations majeures est le fait que les algorithmes peuvent s'enchaîner les uns à la suite des autres, produisant ainsi des solutions de plus en plus efficaces. Par exemple, on pourrait appliquer un algorithme tel qu'AiNet avant d'affiner les résultats par K-means. L'aspect le plus important est qu'il n'y a pas de limite imposée par le logiciel sur le nombre ou les combinaisons particulières d'algorithmes qu'il est possible d'effectuer. Si un algorithme répond à la description ci-dessus et se conforme aux conventions de programmation qui seront détaillées par la suite, il pourra être combiné avec tous les autres algorithmes. La figure 5.9 schématise le système d'hybridation d'algorithmes proposé.

5.5 Adaptation des Algorithmes

Dans cette section, nous présentons les algorithmes de classification de pixels implémentés dans notre plateforme de segmentation. Dans un premier temps, nous présentons un algorithme d'optimisation par essaim de particules adapté pour la classification de pixels. Ce dernier est comparable aux algorithmes k-means et Fuzzy-c-means qui nécessitent la spécification, au préalable, du nombre de classes. Cet algorithme est généralement hybridé avec un autre (ou plusieurs autres) pour améliorer le partitionnement.

Dans un second temps nous présentons l'algorithme Ainet. Comme déjà mentionné, cet algorithme ne requière pas la spécification du nombre de classes au préalable.

5.5.1 Optimisation par essaims de particules (PSO) pour la segmentation d'images

Dans cette section un algorithme de classification inspiré de la méthode d'optimisation par essaims de particules est présenté. Le but principal de cet algorithme, est de servir de méthode de segmentation d'images par classification de pixels.

En appliquant les règles de base de l'optimisation par essaims de particules, l'algorithme décèle les centroids des classes dont le nombre est prédéfini par l'utilisateur. L'efficacité de l'algorithme va être comparée par la suite avec des méthodes de classification similaires, en l'occurrence, l'algorithme du K-means et le Fuzzy-C-means.

5.5.1.1 Description de l'algorithme

Dans le contexte de la segmentation par classification de pixels, chaque particule de l'essaim représente une solution de classification avec K centroids de l'image. Ainsi, chaque particule x_i est définie de la façon suivante :

$$x_i = (m_{i,1}, m_{i,2}, \ldots, m_{i,k}) \tag{5.1}$$

Avec :
$m_{i,j}$ qui représente le j-ième centroids de la particule x_i.

Par conséquent, l'essaim représente les solutions candidates pour le partitionnement des pixels de l'image.

La qualité de chaque particule est mesurée par :

$$f(x_i, Z_i) = w_1 d_{max}(Z_i, x_i) + w_2(z_{max} - d_{min}(x_i)) \tag{5.2}$$

Avec :

- z_{max} est la valeur la plus élevée des niveaux de gris, dans notre cas $z_{max} = 2^8 - 1 = 255$;
- Z_i représente la matrice de distribution des pixels sur les classes de la particule x_i : Chaque élément de la matrice $z_{i,j,k}$ montre si le pixel z_k appartient à la classe C_j dans la particule x_i ;
- w_1 et w_2 sont des variables définies par l'utilisateur agissant sur les deux termes de la fonction de qualité ci-dessus.

Le premier terme de la fonction de mesure $f(x_i, Z_i)$: $d_{max}(Z_i, x_i)$, est défini comme suit :

$$d_{max}(Z_i, x_i) = \max_{j=1,\ldots,K} \{ \sum_{\forall z_k \in C_{i,j}} \frac{d(z_k, m_{i,j})}{n_{i,j}} \} \tag{5.3}$$

Cette formule correspond au maximum des moyennes des distances entre chaque pixel et le centroid de sa classe. Le terme est défini comme suit :

$$d_{min}(x_i) = \min_{\forall k,p=1,\ldots,K,\ k \neq p} \{ d(m_{k,i}, m_{p,i}) \} \tag{5.4}$$

Et correspond au minimum des distances interclasses.

$n_{i,j}$ Représente le nombre de pixels dans la classe C_i selon le partitionnement de la particule x_i.

L'application de la fonction de mesure définie par l'équation 5.2 a pour objectif de :

- Minimiser la distance intra-classe entre les pixels et leur centre ;
- Maximiser les distances interclasses entre les centroids des classes.

D'après la définition de la fonction de mesure, une valeur minimale de $f(x_i, Z_i)$ signifie que les classes sont compactes et bien-séparées.

La fonction de fitness mesurant la qualité d'un bon partitionnement des pixels est une fonction multi-objective, et la méthode de résolution de telles équations se traduit souvent par l'utilisation de méthodes métaheuristiques, dont l'algorithme d'optimisation par essaims de particules. L'objectif sera d'utiliser l'algorithme d'optimisation par essaims de particules pour faire évoluer les solutions représentant des classifications candidates des pixels, et d'estimer, à chaque étape de l'évolution de l'essaim, les solutions en calculant simplement leur poids à l'aide de l'équation 5.2.

Dans ce qui suit, nous allons présenter les différentes étapes de l'algorithme d'optimisation par essaims de particules pour la classification des pixels.

L'avantage principal du PSO par rapport à d'autres méthodes tel le K-means, est que l'algorithme peut s'exécuter en parallèle sur différentes parties de l'image.

Algorithme 7 Algorithme d'optimisation par essaims de particule pour la classification des pixels

Initialiser chaque particule de l'essaim avec K centroids aléatoires
pour $it = 1$ à it_{max} **faire**
  pour chaque particule x_i faire **faire**
    pour tout niveau de gris zp de l'image **faire**
      Calculer $d(z_p, m_{i,j})$ pour toutes les classes $C_{i,j}$
      si $d(z_p, m_{i,j}) = \min_{\forall k=1,\ldots,K} d(z_p, m_{i,k})$ **alors**
        Assigner z_p à $C_{i,j}$
      finsi
      Calculer la fonction de fitness $f(x_i, Z_i)$
    fin pour
    Trouver la meilleure position locale de chaque particule, ainsi que la meilleure position globale de l'essaim
    Mettre à jour les centroids de chaque particule
  fin pour
fin pour

Un second avantage est que l'initialisation des particules n'a pas d'incidence déterminante sur la qualité des résultats notamment lorsque le nombre de particules de l'essaim est important.

5.5.2 AiNet pour la segmentation d'images

Dans cette section nous décrivons comment un algorithme de classification tel qu'AiNet peut être appliqué au problème de la segmentation d'images. Comme précédemment mentionné, le système immunitaire des vertébrés a inspiré de nombreuses théories et paradigmes tels que la théorie des réseaux immunitaires, la sélection par clonage et le principe de maturation d'affinité, qui sont applicables dans le contexte de la classification de données.

AiNet est un algorithme de classification inspiré du fonctionnement du système immunitaire des vertèbres. Il considère les données à classifier comme des antigènes qui viendront stimuler un répertoire d'anticorps. Chaque anticorps réagit à la stimulation de façon différente, en fonction de son degré de correspondance (similarité) avec l'antigène.

Les anticorps présentant une bonne correspondance auront tendance à se multiplier et à proliférer alors que les anticorps n'ayant pas réagit à un antigène depuis un certain temps auront tendance à disparaître. C'est le principe de sélection par clonage.

Le répertoire d'anticorps peut aussi subir des mutations dirigées afin de pouvoir mieux répondre aux stimulations futures : plus la correspondance entre l'anticorps

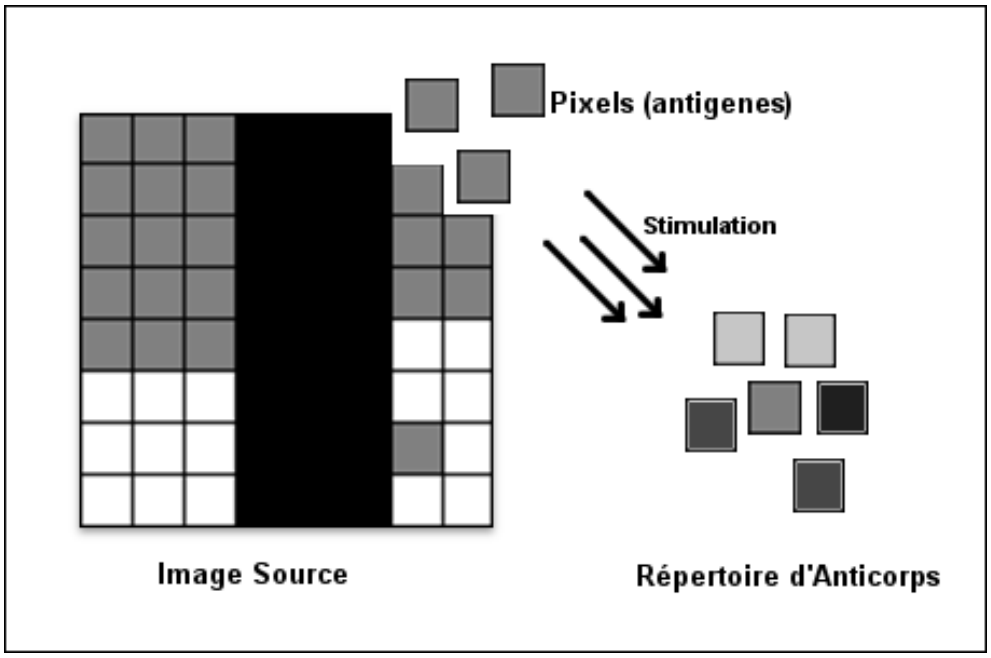

FIG. 5.10 – Modification d'AiNet pour la segmentation d'images.

et l'antigène est grande moins la mutation sera importante et inversement. C'est le principe de maturation d'affinité.

De plus, et afin de limiter la redondance dans le répertoire d'anticorps, les anticorps peuvent réagir à d'autres anticorps. Lorsque deux anticorps correspondent suffisamment, une opération d'élimination a lieu et seulement un des deux est conservé (théorie des réseaux immunitaires).

Après que la stimulation ait été effectuée, le répertoire d'anticorps représentera une « image interne » des antigènes auxquels il a été exposé. Ces anticorps serviront de base pour déduire la classification résultante.

AiNet est un algorithme général de classification de données. Avant de pouvoir l'appliquer à un problème particulier, en l'occurrence la segmentation d'images, il est nécessaire de trouver une modélisation adéquate du problème.

On peut considérer le problème de la segmentation comme un problème de classification des pixels d'une image. Chaque pixel est représenté par son niveau de gris. Dans notre solution, les antigènes et les anticorps seront modélisés par des pixels. L'ensemble des pixels de l'image forme ainsi l'ensemble des antigènes auxquels sera exposé le système. Au début, le répertoire des anticorps est vide. Durant le déroulement d'une passe, le répertoire d'anticorps est exposé aux pixels de l'image (les antigènes) un par un. L'affinité entre les antigènes (pixels de l'image) et les anticorps (pixels du répertoire) est définie comme étant l'inverse de la différence entre leurs niveaux de gris. Au fur et à mesure de l'évolution du répertoire, les anticorps

refléteront l'image de la distribution des pixels sur l'axe des niveaux de gris. Une fois le nombre de passes choisi effectué, chaque pixel du répertoire représente le centre d'une classe de pixels. Autrement dit, le répertoire contient les centroids de la classification résultante. Afin de pouvoir distinguer les pixels de l'image (les antigènes)

Algorithme 8 Algorithme Ainet pour la classification des pixels

pour $T = 1$ à T_{max} **faire**
  pour tout pixel de l'image Ag_i **faire**
    (a) Déterminer son affinité $f_{i,j}$ $i = 1, \ldots, N$ pour chaque pixel Ab_j du répertoire $f_{i,j} = 1/D_{i,j}$ $i = 1, \ldots, N$, $D_{i,j} = \|Ab_i - Ag_i\|$ $i = 1, \ldots, N$
    (b) Un sous-ensemble $Ab\{n\}$ composé des n pixels dont l'affinité est la plus élevée est choisie
    (c) Les n pixels choisis vont proliférer (se cloner) proportionnellement à leur affinité $f_{i,j}$, produisant un ensemble C de clones : plus l'affinité est importante, plus le nombre de clones pour chaque pixel est grand (voir l'équation 5.5)
    (d) L'ensemble C est soumit à un procédé dirigé de maturation d'affinité (mutation guidée) produisant un ensemble muté $C*$, où chaque pixel k de $C*$ subi une mutation de taux a_k inversement proportionnel à l'affinité antigénique $f_{i,j}$ de son anticorps de parent : plus l'affinité est importante, plus le taux de mutation est faible.
    (e) Déterminer $d_k = 1/D_{k,j}$ le taux d'affinité entre Ag_j et tous les éléments de $C*$: $D_{k,j} = \|Ag_j - C_k*\|$
    (f) A partir de $C*$, re-sélectionner $\zeta\%$ des anticorps dont le facteur d_k est le plus élevé, et les mettre dans une matrice M_j de mémoire clonale
    (g) Eliminez tous les clones de M_j dont l'affinité $D_{k,j} < \sigma_d$
    (h) Déterminez l'affinité $s_{i,k}$ entre les clones mémoire : $s_{i,k} = \|M_{i,j} - M_{k,j}\|$
    (i) Eliminez de la mémoire les clones dont le parametre $s_{i,k} < \sigma_s$
    (j) Combiner la matrice totale de pixels avec les clones M_j* pour Ag_j : $Ab\{m\} = [Ab\{m\}, M_j*]$
  fin pour
  Déterminez l'affinité entre tous les pixels mémoire d'Abm : $s_{i,k} = \|Ab\{m\}_i - Ab\{m\}_j\|$ pour chaque i et j
  (Suppression de réseau) Garder une seule occurrence des couples de pixels pour lesquelles $s_{i,k} < \sigma_s$
  Construire la matrice totale d'anticorps $Ab = [Ab\{m\}, Ab\{d\}]$
fin pour

des pixels du répertoire (les anticorps) nous adoptons la convention de designer les premiers par Ag_i et les seconds par Ab. L'algorithme AiNet tel qu'implémenté dans le cadre de ce projet est présenté par l'algorithme 8.

L'équation qui a été utilisée pour calculer le nombre de clones résultant d'une

reconnaissance est la suivante :

$$N_c = \sum_{i=1}^{n} entier(N - D_{i,j} \times N) \tag{5.5}$$

Où N est le nombre total de pixels dans le répertoire d'anticorps, $entier()$ retourne la partie entière de son argument, et $D_{i,j}$ est l'affinité avec laquelle le pixel a été reconnu.

5.6 Evaluation supervisée des résultats

En plus des techniques d'évaluation non supervisée mentionnées dans le chapitre traitant de l'optimisation, l'application développée dans le cadre de ce projet inclut une méthode supervisée d'évaluation. Le principe des méthodes supervisées est de comparer le résultat de l'application de l'algorithme de segmentation avec une segmentation de référence. Une segmentation de référence représente la classification idéale de l'image en question.

Dans le cas des images médicales, en l'occurrence les images IRM, ceux sont les experts du domaine qui se chargent de déterminer la classification correcte ainsi que de fournir la segmentation de référence. Celle-ci prend souvent la forme d'une image de même taille que l'image originale où les pixels ont été recolorés en fonction de leur classification (les pixels de la même classe sont de la même couleur, les pixels de classes différentes sont de couleurs différentes). L'image de référence suit donc le même format que la sortie des algorithmes de segmentation eux-mêmes.

Le problème qui se pose est donc de comparer le résultat de l'application d'un algorithme de segmentation avec une segmentation de référence fournie par un expert du domaine.

L'approche utilisée ici consiste à rechercher pour chaque classe de la segmentation de référence, la classe de la segmentation automatique qui lui correspond le mieux. Cela part du principe que si l'algorithme a extrait chacune des classes de la segmentation de référence alors le résultat doit être considéré comme satisfaisant. Le résultat global de la comparaison est alors la somme des distances entre les couples de classes correspondant le mieux.

Pour ce faire, un critère de comparaison permettant de mesurer la similarité entre deux classes de pixels dans deux segmentations différentes est nécessaire. C'est le critère de *Jaccard* qui à été utilisé dans notre projet. Ce dernier calcule le nombre de pixels dans l'intersection des classes et le nombre de pixels dans l'union, puis effectue la division des deux résultats. L'algorithme de calcul de la distance de Jaccard entre

deux classes de pixels est montré dans l'algorithme 9.

Algorithme 9 Critère de Jaccard pour mesurer la similarité entre deux classes de pixels $C1$ et $C2$ de deux segmentations d'une même image I.

Soit $P_{inter} = 0$, $P_{union} = 0$
pour tout pixel z_p de l'image I **faire**
  si $z_p \in C1$ ET $z_p \in C2$ **alors**
    $P_{inter} = P_{inter} + 1$
  sinon si $z_p \in C1$ OU $z_p \in C2$ **alors**
    $P_{union} = P_{union} + 1$
  finsi
fin pour
Retourner P_{inter}/P_{union}

L'algorithme qui permet de comparer le résultat d'un algorithme de segmentation avec la segmentation de référence est présenté dans l'algorithme 10.

Algorithme 10 Algorithme de comparaison entre le résultat d'une segmentation automatique I_{seg} et sa réference I_{ref}

Mettre la variable $Crsp$ à 0
pour Chaque classe C_{ref} de I_{ref} **faire**
  pour Chaque classe C_{seg} de I_{seg} **faire**
    Calculer le critère de Jaccard entre C_{seg} et C_{ref}
    Soit C'_{seg} la classe de I_{seg} dont l'indice de Jaccard est le plus élevé
    $Crsp = Crsp + critere_de_jaccard(C_{ref}, C'_{seg})$
  fin pour
fin pour
Retourner $Crsp$

5.7 Mise en œuvre

Le greffon développé dans le cadre de ce projet se compose de deux programmes distincts correspondant aux deux modes opératoires disponibles :

Mode image unique : Le principe est d'appliquer les algorithmes de segmentation à une image à la fois afin de mieux examiner et ajuster les résultats pour une image particulière. L'objectif étant dans ce cas d'obtenir les meilleurs résultats sur une image donnée.

Mode « batch » : Dans le « mode batch » il s'agira de choisir une combinaison particulière d'algorithmes, d'ajuster leurs paramètres puis de les appliquer à un grand nombre d'images avant de récupérer un résumé des résultats. L'objectif

étant dans ce cas de tester le comportement des algorithmes sur une grande quantité de données.

Ces deux modes sont en fait complémentaires : on pourrait utiliser le mode image unique pour ajuster au mieux les paramètres d'un algorithme avant de tester l'efficacité de ce paramétrage sur un grand nombre d'images.

Les sections suivantes constituent un manuel d'utilisation : elles détaillent l'utilisation de l'interface ainsi que l'interprétation des résultats de segmentations dans les deux modes opératoires.

5.7.1 Mode Image Unique

Le mode image unique opère sur une image particulière qui doit donc être ouverte au préalable dans GIMP. L'installation du greffon provoque l'ajout d'un menu « Segmentation » dans toutes les images ouvertes. C'est ce menu qui invoque le mode image unique de l'application (voir figure 5.11).

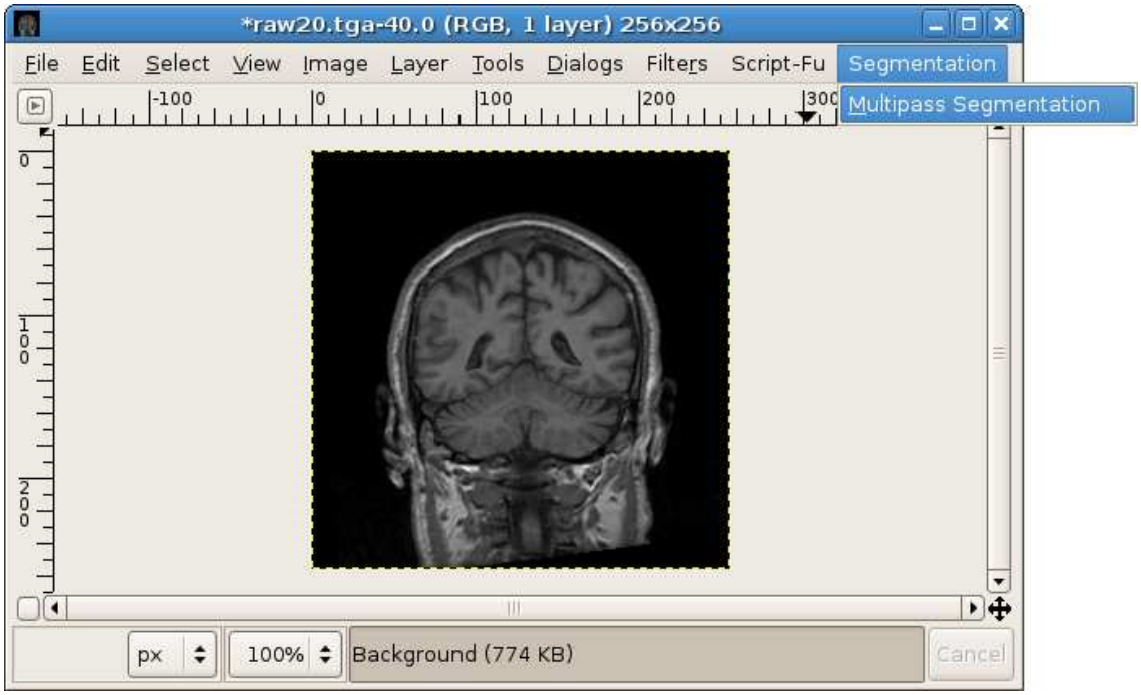

FIG. 5.11 – Le menu segmentation déclenche le mode image unique.

Les algorithmes de segmentation considérés ici opèrent sur des images en niveaux de gris. Si l'image traitée est une image couleur (RGB, ou couleurs indexées) la conversion en niveaux de gris se fera automatiquement. Cependant, cette conversion automatique ne donne pas toujours les meilleurs résultats et il est donc recommandé d'effectuer l'opération de conversion en utilisant les outils disponibles dans GIMP.

Une fois le menu sélectionné l'interface principale du mode image unique apparaît (Voir figure 5.12).

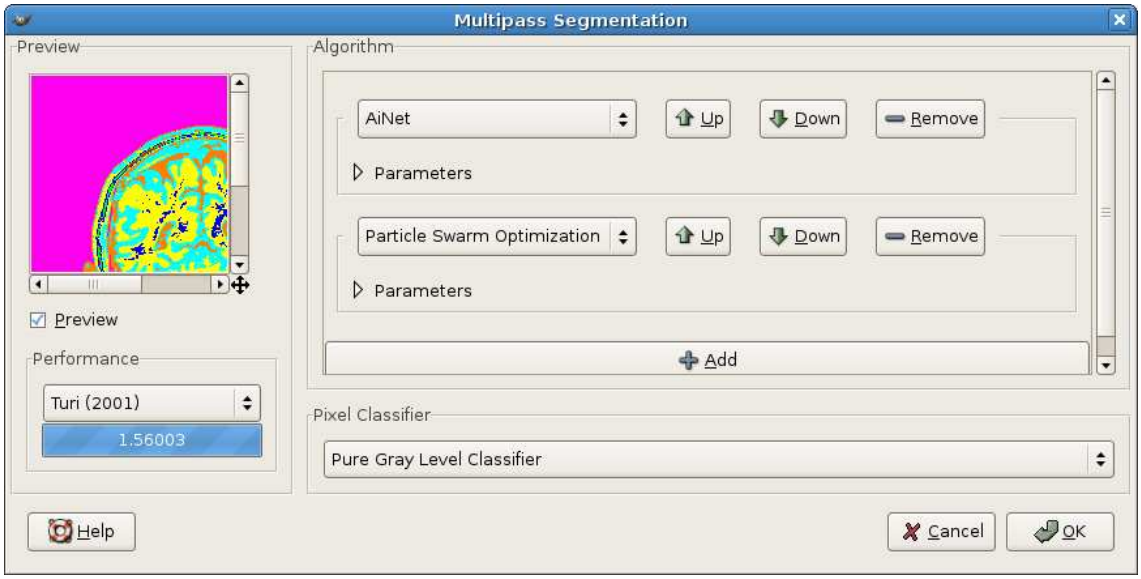

FIG. 5.12 – L'interface principale du mode image unique.

L'interface du mode image unique permet la sélection des algorithmes de segmentation à appliquer ainsi que l'ajustement des paramètres. La fenêtre se compose de plusieurs panneaux remplissant chacun un rôle prédéfini.

Le panneau « Algorithm » (en haut à droite) est le panneau principal de l'interface. Il permet de sélectionner la combinaison particulière d'algorithmes à appliquer. Il s'agit en fait d'une liste d'algorithmes qui seront appliqués de haut en bas, successivement. Un bouton « Add » positionné en bas de la liste permet d'ajouter un algorithme à la fin de la séquence.

Chaque algorithme de la liste est représenté par un sous-panneau comprenant le nom de l'algorithme, trois boutons : « Up », « Down » et « Remove », ainsi que la mention « Parameters ». Il est possible de changer à tout moment l'algorithme en question en cliquant sur son nom. Une liste de choix des algorithmes de segmentation disponibles est affichée.

En fonction de l'algorithme sélectionné dans un sous panneau, plusieurs paramètres pourront être ajustés. Ces paramètres sont, par défaut, masqués sous la mention « Parameters » dans le sous panneau. Pour les afficher (et ainsi pouvoir les ajuster), il suffit de cliquer sur le bouton « Parameters » (Voir figure 5.13).

Le deuxième panneau, à gauche de la fenêtre (titré « Preview ») permet d'obtenir un aperçu des résultats de la séquence d'algorithmes sélectionnée sur une portion de l'image.

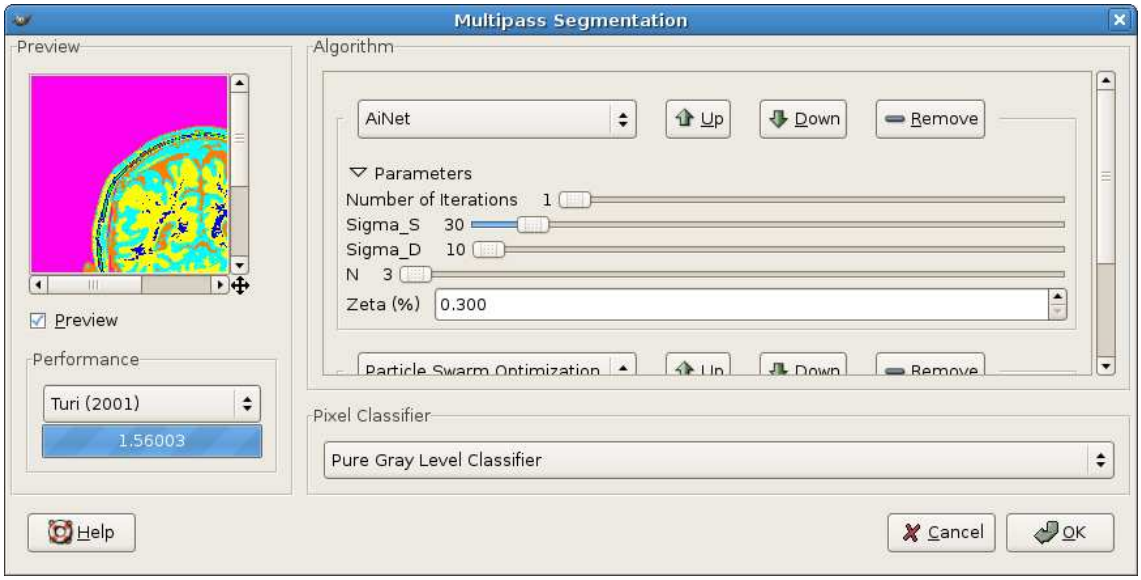

FIG. 5.13 – Ajustement des paramètres

La fenêtre d'aperçu elle-même montre une portion de l'image à laquelle a été appliquée la segmentation. L'image est recolorée pour mettre en valeur les différentes classes de pixels (les pixels de la même classe reçoivent la même couleur).

L'aperçu est généré de façon continue au fur et à mesure que la séquence d'algorithmes est modifiée et que les paramètres sont ajustés. Si la surcharge induite est trop importante, il est possible de désactiver la mise à jour de l'aperçu en cliquant sur le bouton à deux états « Preview ».

En bas du panneau « Preview » se trouve un sous panneau « Performance », qui donne une estimation de la qualité de la segmentation sur la portion de limage en cours de traitement (à savoir l'aperçu). Cette estimation est en fait le résultat de l'application d'une des fonctions de fitness disponibles. Il est possible de choisir la fonction de fitness désirée en cliquant sur la liste déroulante affichant la sélection en cours. L'aperçu ainsi que l'estimation des résultats ne concernent qu'une portion de l'image. Le résultat de l'application de l'algorithme à l'image entière peut d'être différent (qualité supérieure ou moindre).

Si l'image est enregistrée sur le disque sous un nom tel que « `fichier.ext` » et qu'une image de même taille est enregistrée dans le même répertoire sous le nom de « `fichier_seg.ext` », alors cette dernière est considérée comme une segmentation de référence pour l'image en cours. Dans ce cas, la liste des fonctions de fitness comporte une entrée supplémentaire : « Jaccard », qui donne une estimation supervisée de la segmentation. Cette convention de nommage de la segmentation de référence reste valide pour la fenêtre des résultats, ainsi que dans le mode batch.

Le dernier panneau (en bas de la fenêtre) permet de choisir entre deux méthodes

d'affectation de pixels aux classes :

- La méthode classique, ou chaque pixel est affecté à la classe dont le centre est le plus proche ;
- La méthode basée sur la position où le voisinage de chaque pixel influence son affectation.

Après avoir choisi les algorithmes, ajusté les paramètres, et vérifié l'aperçu, il ne reste plus qu'a cliquer sur « OK » pour démarrer la segmentation elle-même.

Compte tenu de l'implémentation adoptée, l'opération ne devrait pas durer plus de quelques secondes même pour de grandes images (1024x768 pixels, ou même 1600x1200 pixels).

Une fois la segmentation terminée, deux fenêtres apparaissent :

- L'image segmentée : qui est une copie de l'image originale dont les pixels ont été recolorés pour refléter leur classification.
- un résumé de statistiques sur la classification : Les données sont regroupées à chaque étape de la segmentation et sont présentées séparément. La fenêtre contiendra donc un tableau de statistiques résumant l'état des classes après chaque étape.

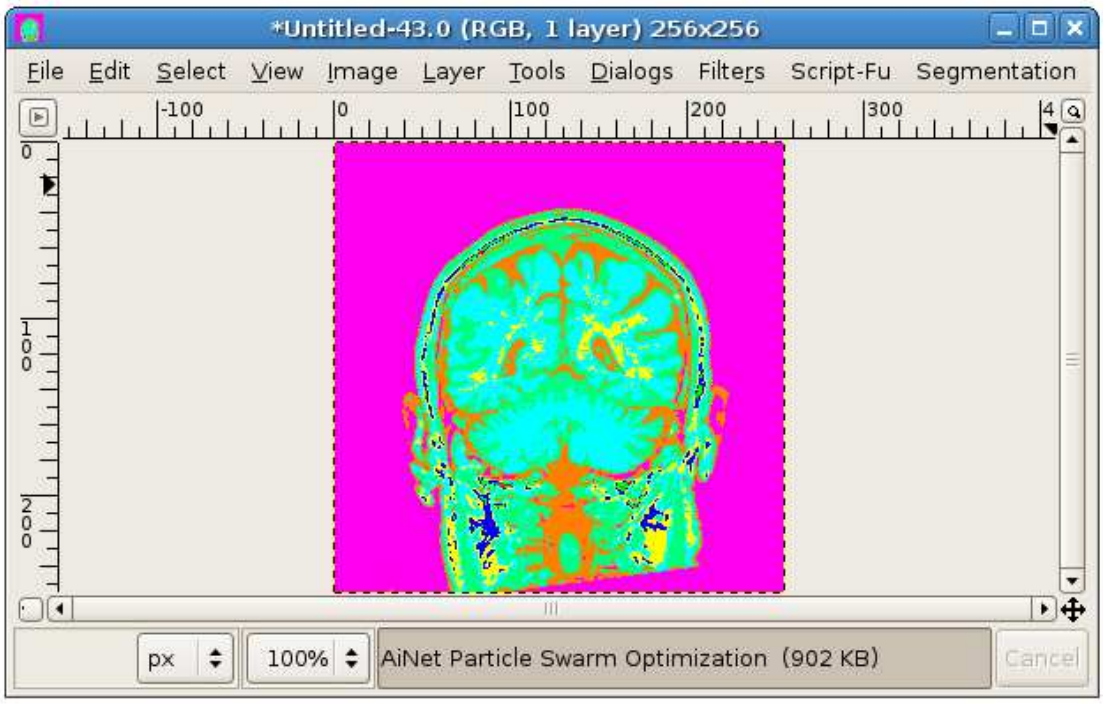

FIG. 5.14 – Image après segmentation.

Par défaut, seuls les résultats finaux (correspondant à la dernière étape) sont affichés, les autres sont masqués. Pour les afficher, il suffit de cliquer sur la mention

« Results » correspondante.

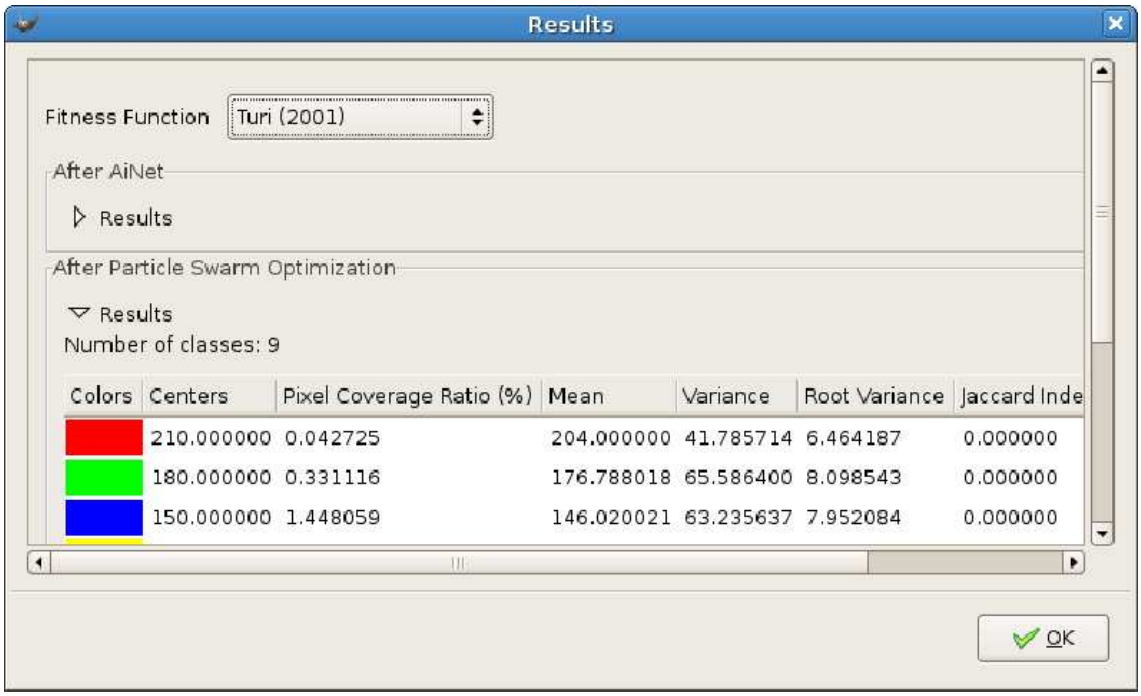

FIG. 5.15 – Données statistique sur la segmentation.

Pour chaque étape, la fenêtre affiche le nombre de classes ainsi que pour chaque classe des informations telles que :

- La couleur (qui permet de retrouver la classe dans l'image segmentée) ;
- La valeur du centroid correspondant ;
- Le pourcentage de pixels appartenant à la classe (« Pixel Coverage Ratio ») ;
- La moyenne (« Mean ») ;
- La variance (« Variance ») et L'écart type des niveaux de gris des pixels dans la classe.

La fenêtre affiche aussi une estimation de la qualité de la classification telle que fournit par une fonction de fitness. Il est possible de choisir la fonction de fitness en question (parmi celles disponibles) en cliquant sur la liste déroulante en haut de la fenêtre.

5.7.2 Mode «Batch»

Le mode batch n'opère pas sur une image en particulier mais sur un grand nombre d'images à la fois. C'est pourquoi le sous menu invoquant cette fonctionnalité n'est pas dans le menu « Image » de GIMP mais dans le menu principal (au dessus de la boite à outils, voir la figure 5.16).

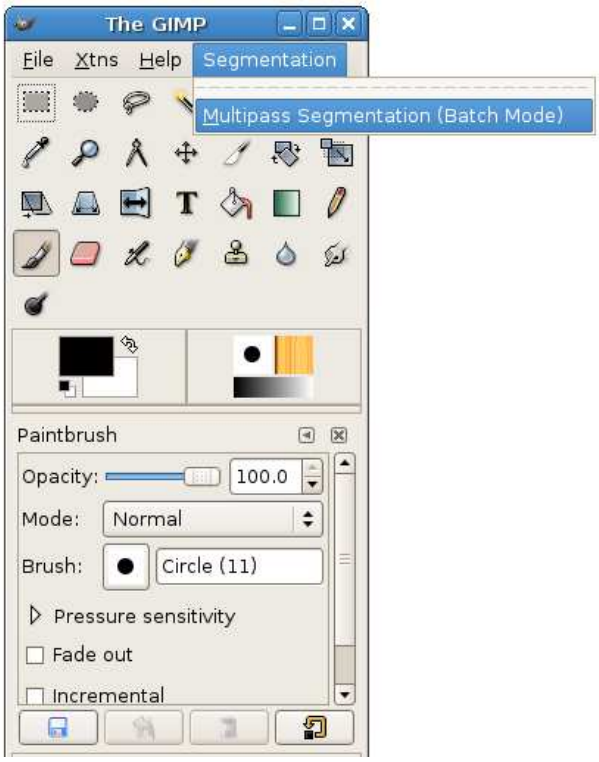

FIG. 5.16 – Menu du mode "batch"

Une fois ce menu invoqué, la fenêtre de la figure 5.17 apparaît. L'interface du mode batch est composée de deux parties :

- la partie inférieure qui représente le panneau « Benchmark». Elle est constituée de deux zones : la zone « Save results to » et la zone « Files ». La première doit contenir le nom du fichier dans lequel les résultats de la segmentation seront stockés. La deuxième zone contient les noms des fichiers images à segmenter. Au dessus de la zone correspondant au nom des fichiers à traiter, quatre boutons y figurent : le bouton « Add » qui permet d'ajouter une image dans la liste à traiter, le bouton « Remove » qui permet d'ôter une image de la liste, le bouton « Save » qui permet de sauvegarder la liste, et le bouton « Open » pour charger une liste d'images à traiter.
- la partie supérieure qui représente le panneau « Algorithm ». Ce dernier est similaire à celui présenté précédemment, il permet d'éditer la liste des algorithmes à appliquer et d'ajuster leurs paramètres ;

La dernière étape consiste à choisir un fichier où seront sauvegardés les résultats, soit en entrant directement *son nom* dans la zone « Save results to », ou en parcourant les disques durs à la recherche d'un emplacement en cliquant sur le bouton «...».

Les résultats d'une segmentation en mode batch consistent en un fichier texte

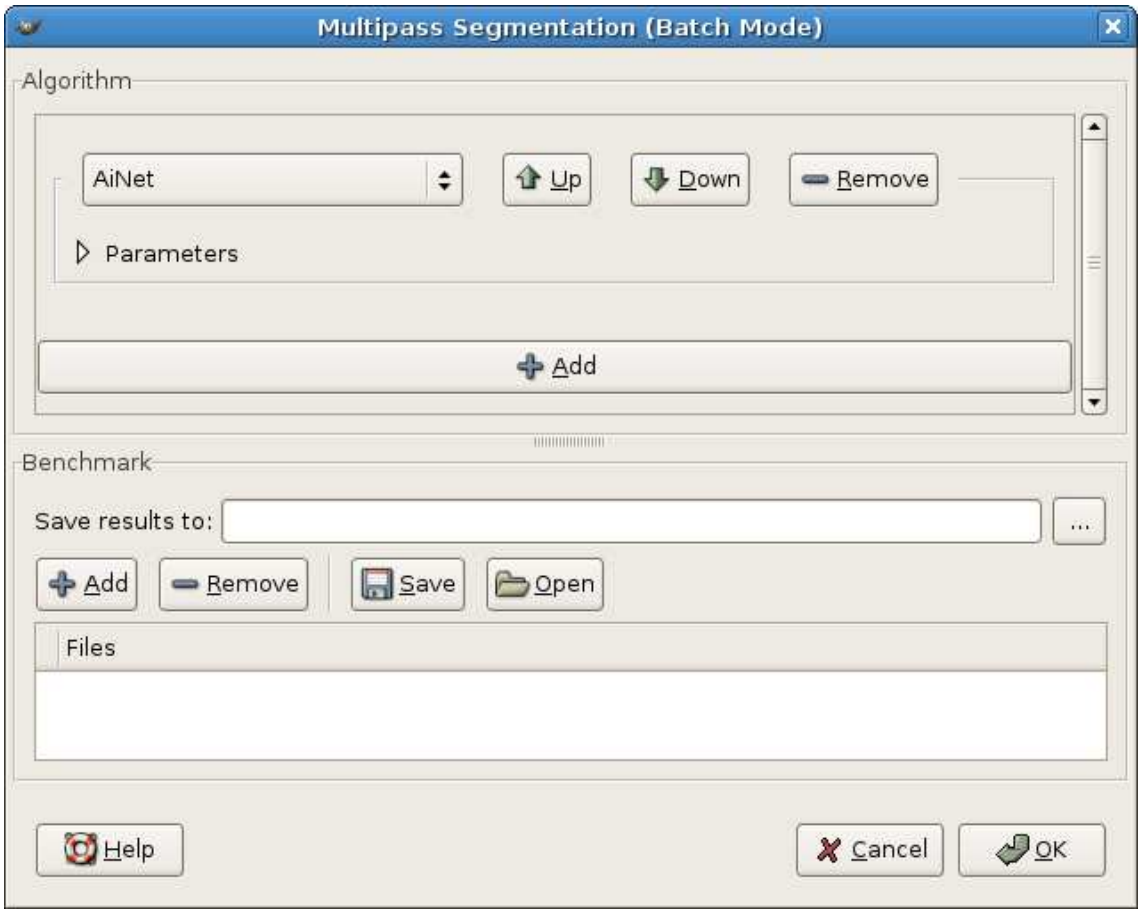

FIG. 5.17 – Interface du mode batch.

résumant des données statistiques correspondantes à l'opération dans son ensemble.

Pour chaque image, le fichier contiendra le nombre de classes résultantes, le temps d'exécution, ainsi que le résultat de l'application des fonctions de fitness. Ce fichier pourra alors aisément être introduit comme entrée pour un tableur à partir duquel pourront être produits des graphes permettant une lecture claire et précise des résultats obtenus et des comparaisons de résultats de plusieurs applications.

5.8 Extension de l'application

Les paragraphes qui suivent détaillent le schéma de programmation suivi par les algorithmes. Ils décrivent donc les étapes à suivre pour étendre l'application et ajouter de nouveaux algorithmes.

Dans le cadre de ce travail, un algorithme de segmentation particulier dans le contexte d'un enchaînement d'algorithmes est appelé une passe. Mise à part une solution initiale et un histogramme, une passe aura souvent besoin de plusieurs autres paramètres permettant de personnaliser son comportement (par exemple, un seuil

de précision, ou le nombre de classes initiales). Ces paramètres ne sont pas universels et dépendent de la passe particulière considérée. Il relève donc de la responsabilité de la passe elle-même de déclarer le nombre et le type de paramètres qu'elle requière pour que l'application puisse générer l'interface utilisateur permettant d'ajuster ces paramètres, et ainsi permettre à la passe elle-même de consulter les valeurs choisies par l'utilisateur.

Afin d'illustrer cette notion, nous décrirons une passe modèle, qui contient tout le code nécessaire excepté l'algorithme de classification lui-même. Cette passe modèle se trouve implémentée dans les fichiers `template_pass.hpp` et `template_pass.cpp`. Ces fichiers peuvent donc servir de point de départ pour le développement et l'intégration de nouveaux algorithmes dans l'application.

Noter que dans ce paragraphe nous aborderons certaines classes et méthodes faisant partie du noyau de l'application. Cependant, ce document n'étant en aucun cas un substitut de la documentation présente dans le code source lui-même, il est recommandé au lecteur de se référer au code source et principalement aux fichiers d'entête pour plus de détails.

5.8.1 Le fichier d'entête

Le seul objet de `template_pass.hpp` est de déclarer la classe `template_pass` :

```
#include "pass.hpp"
/// A template for implementing a new pass
class template_pass: public pass
{
        private:
                /// The number of parameters we need
                enum { m_param_info_size=2 };

                /// Holds a param_info for each parameter we take
                static param_info m_param_info[m_param_info_size];
}
```

Le fichier commence par déclarer la classe `template_pass` comme une classe fille de la classe `pass`. Il est aisé de déduire que la classe pass (c.f. fichier `pass.hpp`) est la parente de toutes les classes représentants une passe.

L'étape suivante est d'informer la classe `pass` du nombre et du type de paramètres que `template_pass` requière. Ceci est obtenu en constituant un tableau d'objets de type `param_info`. Un objet de type `param_info` (c.f. `param_info.hpp`)

contient des informations sur un paramètre d'une passe. Le programme ci-dessus déclare simplement un tableau statique (`m_param_info`) d'objets `param_info` appelé à contenir l'information sur les paramètres de la passe. Nous noterons que ce code porte uniquement sur la déclaration du tableau qui sera rempli à une étape ultérieure.

```
public:
        /// Pass a label, an array of param_info and the size of the
        /// array to parent class
        template_pass():
                pass("Template", m_param_info, m_param_info_size)
        {}
```

Puis vient l'interface public de la classe. Il est à noter que le constructeur de la classe `pass` prend trois arguments :

1. Une étiquette (`char const*`) : Qui servira de nom externe à la passe, et permettra à l'utilisateur de l'identifier parmi les autres. Ce nom doit être unique dans le contexte des autres passes de l'application ;
2. Un tableau d'objets de type `param_info` : Qui contient un objet `param_info` pour chaque paramètre de la passe. Noter comment le code passe le tableau `m_param_info` vue précédemment ;
3. Le nombre d'éléments dans le tableau.

```
        /// The actual implementation of the algorithm
        /// historgram: Holds the number of pixel of each possible
        ///    gray level in the image.
        /// centers: The input class centers. The algorithm should
        ///    update them/improve them.
        /// ind: Can be used to report the progression
        virtual void operator() (histogram const & ,
                        centroid_set& centers,
                        progress_indicator* ind) const;
```

Le code déclare ensuite le membre le plus important. C'est cette surcharge de « l'opérateur parenthèses » (`operator()`) qui va effectuer la segmentation elle-même. Nous constatons aisément que les deux premiers arguments ne sont autres que les deux données en entrée d'un algorithme que nous avons décrites précédemment, à savoir : L'histogramme (c.f. `histogram.hpp`) et un ensemble de centroids (c.f. `centroid_set.hpp`).

Il est également très important de noter que l'histogramme est passé en tant que référence constante, la passe ne peut donc pas le modifier. Cependant, l'ensemble des centroids est passé, lui, en tant que référence non constante. Ce qui signifie qu'il est un argument en entrée (il donne la solution initiale), mais aussi un paramètre en sortie, car il permet à l'algorithme de retourner la solution améliorée directement dans le paramètre lui même. En d'autres termes, l'objectif est de mettre à jour la solution reçue.

Le dernier paramètre n'a pas d'impact sur la segmentation elle-même, mais permet à la passe de renseigner l'application sur le taux d'avancement courant de l'algorithme. Cet argument possède un membre `operator()` que nous pouvons invoquer en passant une estimation du pourcentage de la progression de l'algorithme (0.0 pour aucun travail effectué pour l'instant, 1.0 pour travail terminé). (voir le fichier `progress_indicator.hpp` pour plus de détails).

```
/// Must return a new copy of *this
virtual pass* clone() const { return new template_pass(*this); }
};
#endif // TEMPLATE_PASS_HPP
```

La derrière partie du fichier contient le dernier membre de la classe : `clone()`, dont le rôle est de retourner une copie de l'instance courante allouée avec l'operateur `new`.

5.8.2 Le fichier source

Le rôle du fichier source est de définir les paramètres de la passe et l'algorithme lui-même :

```
#include "template_pass.hpp"
#include "histogram.hpp"
/// Definition of parameter info array
///(contains an element per parameter)
param_info
template_pass::m_param_info[template_pass::m_param_info_size]=
{
   /// First parameter will be an integer from 1 to 10 named
   /// "Number of Iterations"
   param_info(1, 10, "Number of Iterations",
               "Number of passes to apply"),
```

```
   /// Second parameter will be a double between 0 and 1
   param_info(0.0, 1.0, "Precision",
               "Accuracy of calculations")
};
```

Nous notons la définition du tableau `m_param_info` mentionné précédemment et comment le programme le remplit avec des instances de *param_ info* (voir *param_ info.hpp*).

```
// Actual implementation of the algorithm
void template_pass::operator() (histogram const& hist,
                        centoid_set& centroids,
                        progress_indicator* ind) const
{
        // First extract actual parameter values
        int const nr_passes=this->pass::parameter_value(0).
                                        int_value();
        double const precision=this->pass::parameter_value(1).
                                        int_value();
```

Enfin vient l'implémentation de l'algorithme lui-même. La première opération à effectuer est de récupérer les valeurs des paramètres choisis par l'utilisateur. D'après ce qui précède, la passe prend deux arguments : un entier entre 1 et 10, et un réel entre 0.0 et 1.0.

Pour récupérer les valeurs effectives de ces paramètres il suffit d'invoquer la fonction `pass::parameter_value()` (implémentée dans la classe parente : `pass`) en passant l'index du paramètre concerné. La valeur retournée par `parameter_value()` ne peut être simplement un `int` ou un `double`, du fait que l'information sur le type du paramètre ne peut être connue au moment de la compilation. C'est pourquoi la valeur de retour est un objet de type `param` qui est un conteneur de données flexible contenant soit un entier soit un réel. Il reste simplement à invoquer `param::int_value()` pour obtenir un entier (dans le cas de paramètres entiers), ou `param::double_value()` pour obtenir un `réel` (dans le cas de paramètres `réels`).

```
        double total_pixels=0;
        // Count the total number of pixels
        for(histogram::const_iterator it=hist.begin();
                        it!=hist.end(); ++it)
```

```
                total_pixels+=*it;
[
        int pass_count=nr_passes;
        while(pass_count--){
                // Go through the histogram
                for(histogram::const_iterator it=hist.begin();
                      it!=hist.end(); ++it){
                        // do something about the centroids
                }
        }// while pass_count--
}
```

Le reste du fichier est un modèle vide auquel il faudra ajouter un algorithme de segmentation. Dans un exemple réel, le code utilisera l'histogramme et la solution initiale pour améliorer cette dernière.

Après avoir procédé à l'écriture du programme lui-même, il faut l'enregistrer auprès de l'application pour qu'il apparaisse dans la liste des algorithmes disponibles. Pour ce faire, nous devons procéder à l'ouverture du fichier `register_pass.cpp` et apporter des modifications comme indiqué dans le fragment ci-après :

```
#include "register_passes.hpp"
#include "pass_repertoire.hpp"
#include "ainet_pass.hpp"
#include "kmeans_pass.hpp"
#include "template_pass.hpp"  // <<< See here ////////

void register_passes()
{
        pass_repertoire& rep=*pass_repertoire::instance();

        rep.add(new ainet_pass());
        rep.add(new kmeans_pass());

        rep.add(new template_pass()); // <<< See here ////////
}
```

La dernière étape porte sur la recompilation de l'application sans omettre d'y inclure les fichiers nouvellement créés.

5.9 Conclusion

Dans ce chapitre, nous avons présenté les points essentiels de notre application. Nous avons commencé par une description de l'environnement de travail qui à servi de base pour notre plateforme de segmentation. Nous avons ensuite présenté son architecture générale, ainsi que les algorithmes biomimétiques modélisés. Nous avons aussi décrit les différentes étapes à suivre pour ajouter d'autres algorithmes de classification. Enfin, la dernière section constitue un guide d'utilisation de notre application.

Dans le chapitre suivant, nous allons présenter les tests effectués. L'objectif étant de comparer les résultats produits par les algorithmes biomimétiques modélisés, ainsi, que d'étudier l'influence des hybridations sur les résultats de segmentation.

Chapitre 6

Tests et Résultats

6.1 Introduction

Dans ce chapitre, nous présentons les tests effectués et les résultats obtenus après application des différents algorithmes de segmentation développés dans le cadre de ce travail.

Les tests présentés dans ce chapitre ont été réalisés sous le système d'exploitation Debian GNU/Linux version 4.0 «etch», tournant sur un microprocesseur Intel Core 2 Duo (Merom) T5200 avec 1 GO de RAM.

6.2 Benchmarks

Les images utilisées pour les différents tests sont des images médicales acquises par résonance magnétique (IRM). Deux benchmarks ont été utilisés pour effectuer les tests :

Dhawan : 48 images par résonance magnétique du cerveau fournies par Atam Dhawan pour illustrer son ouvrage «Medical Image Analysis» ([Dhawan, 2003]) ;

IBSR : 53 images par résonance magnétique (IRM) fournies par l'«Internet Brain Segmentation Repository»[ibsr, 2007]. C'est un benchmark *supervisé*, ce qui permet de comparer les résultats obtenus aux attentes des praticiens.

La figure 6.1 montre des images extraites du benchmark IBSR, et la figure 6.2 montre quelques images du benchmark de Dhawan.

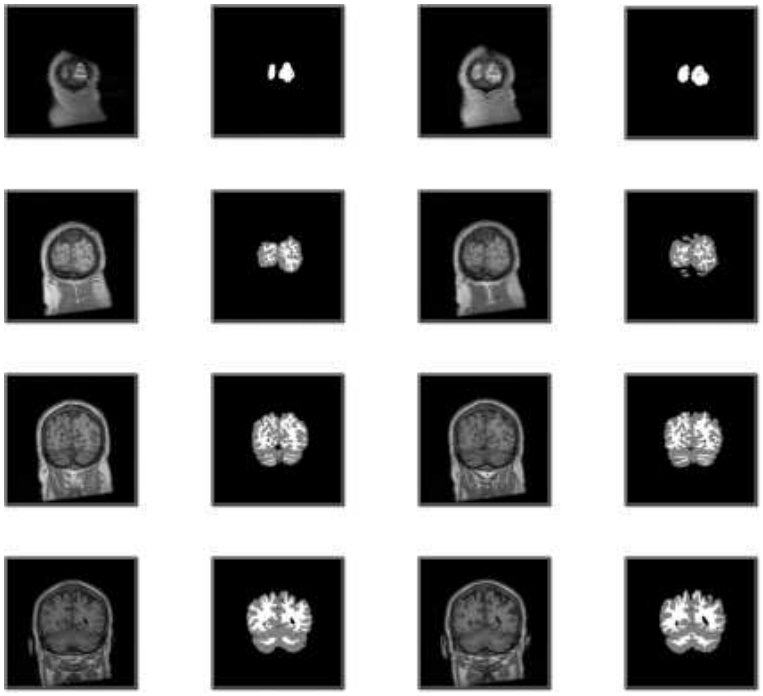

FIG. 6.1 – Images IRM et leur segmentation de référence d'après [ibsr, 2007]

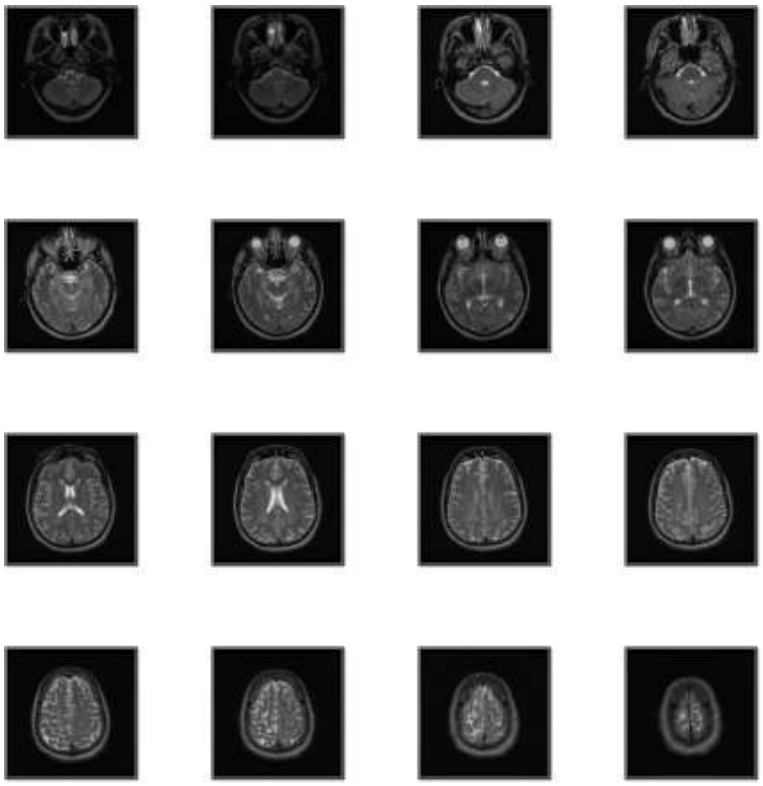

FIG. 6.2 – Ensemble d'images IRM du cerveau [Dhawan, 2003]

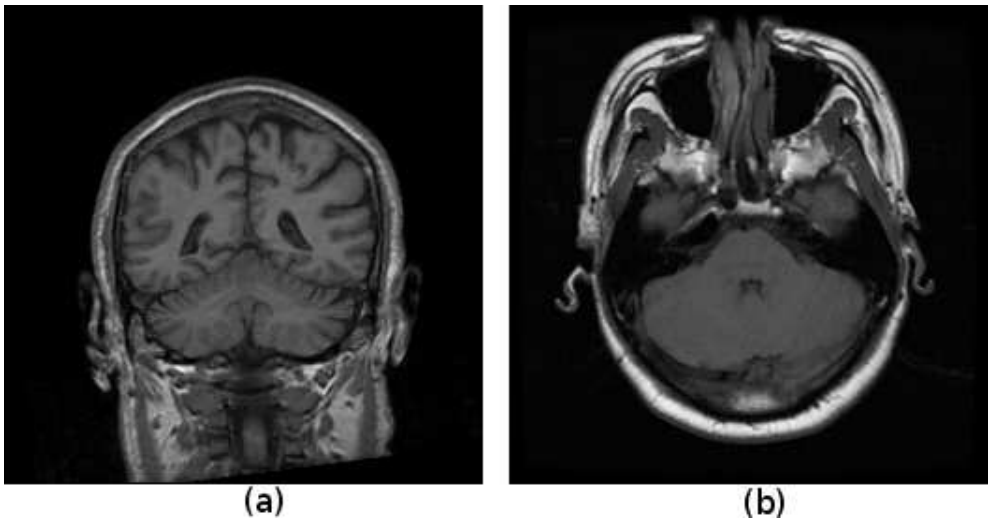

FIG. 6.3 – Les deux images utilisées pour les tests des algorithmes

6.3 Tests par algorithme

Dans cette section, nous présentons les tests effectués sur chaque algorithme de notre plateforme : Ainet, PSO, K-means, et Fuzzy-C-means. Ces différents tests ont été réalisés sur deux images : une image du benchmark IBSR, et une autre du Benchmark Dhawan (voir la figure 6.3). Les résultats obtenus sont présentés dans une perspective de comparaison visuelle. Une étude comparative des performances des algorithmes de segmentation est présentée dans la section 6.4.

6.3.1 Tests sur Ainet

Dans cette partie, nous nous intéressons à un paramètre particulier de *Ainet* : σ_s. Ce paramètre σ_s correspond au seuil de suppression des clones de cellules immunitaires et influe donc sur le nombre de classes générées.

Les figures 6.4 et 6.5 montrent les résultats obtenus pour chaque valeur du paramètre σ_s utilisée à savoir : 10, 35, et 100.

6.3.2 Tests sur PSO

Les tests effectués sur le *PSO* montrent le comportement de l'algorithme lorsque la taille de chaque particule est modifiée. Dans cet algorithme la taille d'une particule détermine le nombre de classes dans l'image segmentée. Les figures 6.6 et 6.7 montrent les résultats obtenus pour chaque valeur du paramètre *particlesize*, à savoir : 7, 10, et 20.

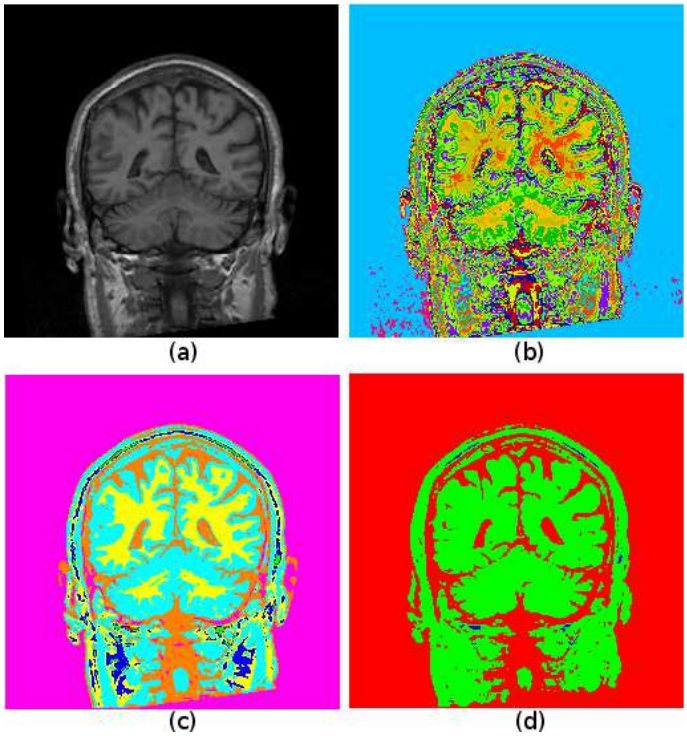

FIG. 6.4 – (a) Image initiale (IBSR). (b) Segmentation par Ainet avec $\sigma_s = 10$. (c) Segmentation avec $\sigma_s = 35$. (d) Segmentation avec $\sigma_s = 100$

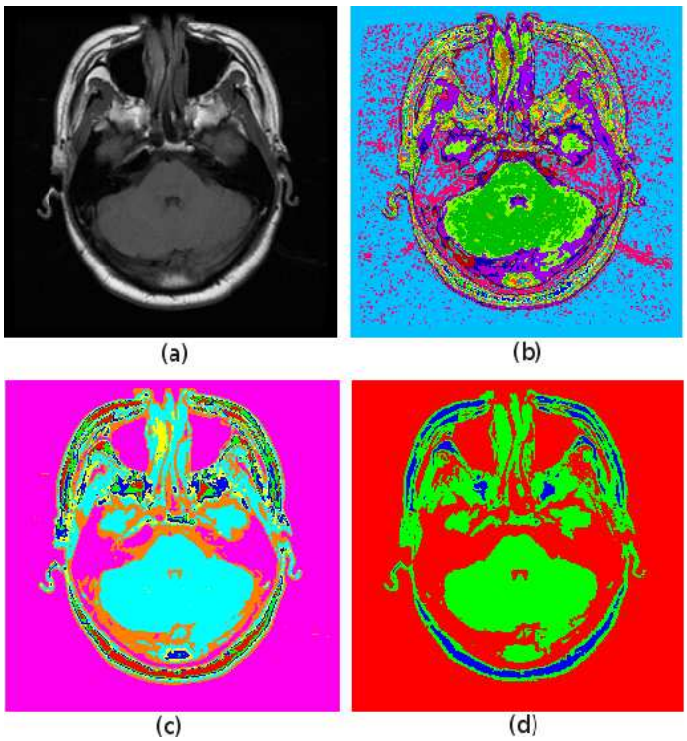

FIG. 6.5 – (a) Image initiale (Dhawan). (b) Segmentation par Ainet avec $\sigma_s = 10$. (c) Segmentation avec $\sigma_s = 35$. (d) Segmentation avec $\sigma_s = 100$

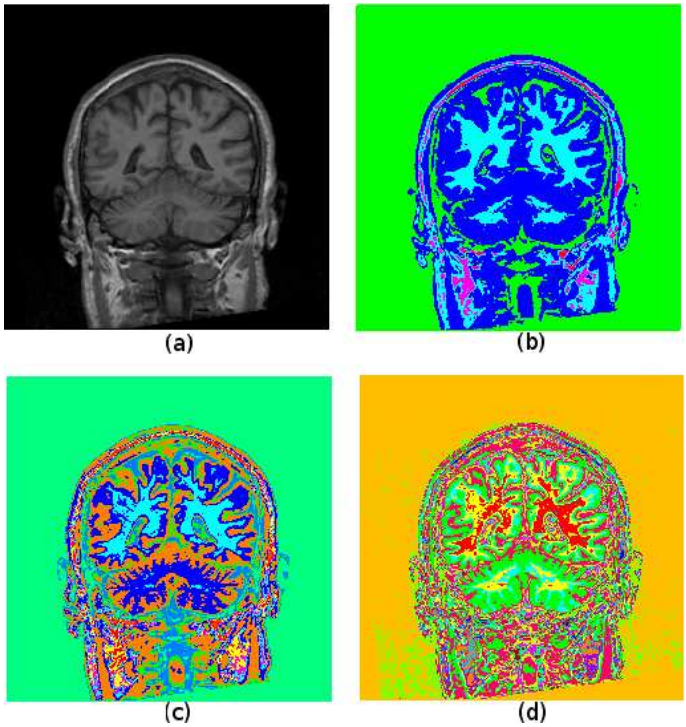

FIG. 6.6 – (a) Image initiale (IBSR). (b) Segmentation par PSO avec *particle size* = 7. (c) Segmentation avec *particle size* = 10. (d) Segmentation avec *particlesize* = 20

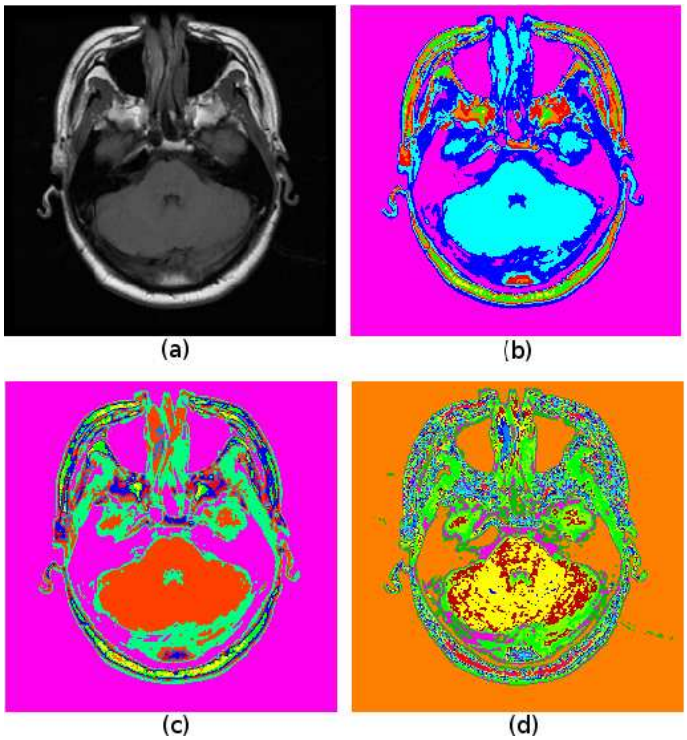

FIG. 6.7 – (a) Image initiale (Dhawan). (b) Segmentation par PSO avec *particlesize* = 7. (c) Segmentation avec *particle size* = 10. (d) Segmentation avec *particlesize* = 20

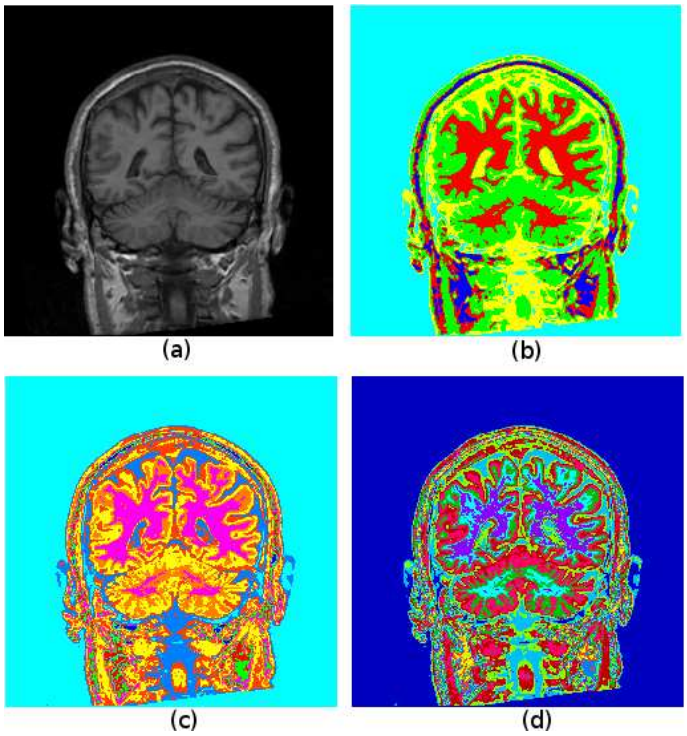

FIG. 6.8 – (a) Image initiale (IBSR). (b) Segmentation par K-means avec $clusters = 5$. (c) Segmentation avec $clusters = 10$. (d) Segmentation avec $clusters = 20$

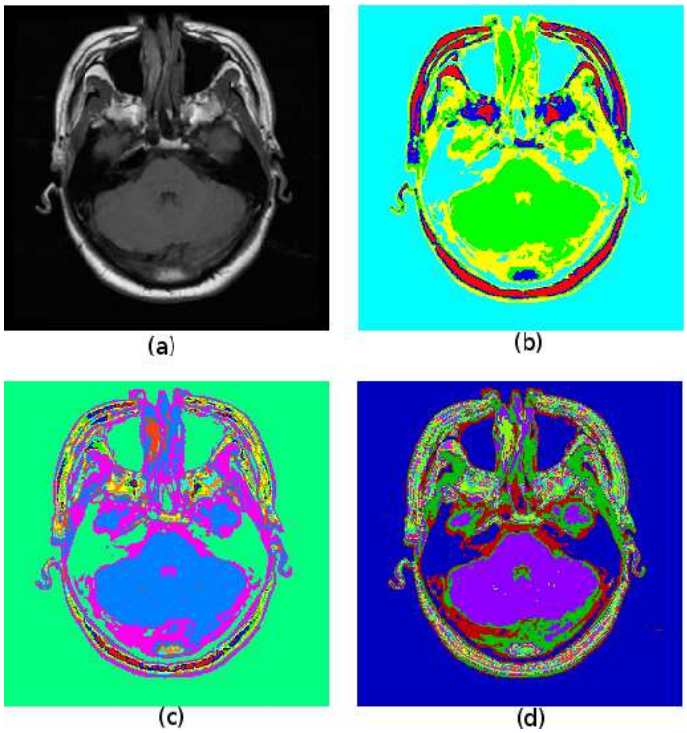

FIG. 6.9 – (a) Image initiale (Dhawan). (b) Segmentation par K-means avec $clusters = 5$. (c) Segmentation avec $clusters = 10$. (d) Segmentation avec $clusters = 20$

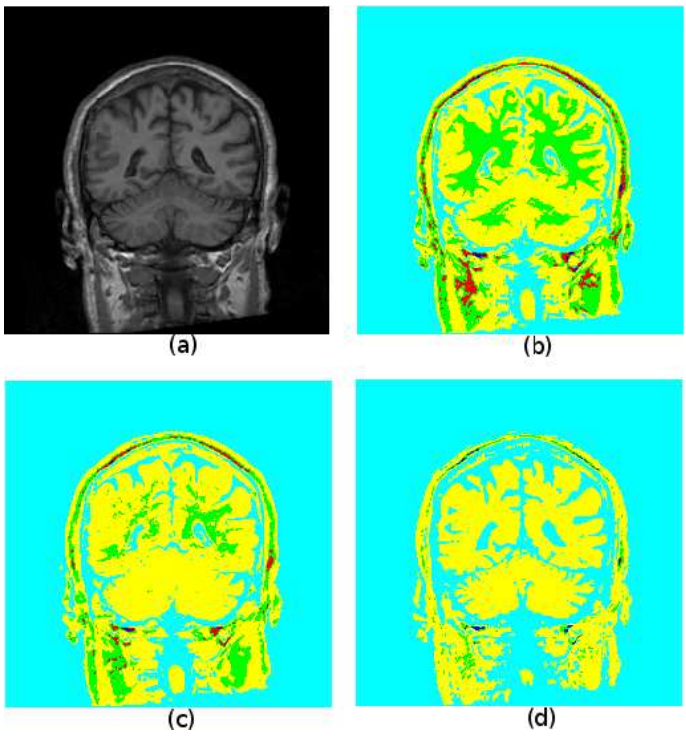

FIG. 6.10 – (a) Image initiale (IBSR). (b) Segmentation par FCM avec $fuzzy\ index = 2$. (c) Segmentation avec $fuzzy\ index = 5$. (d) Segmentation avec $fuzzy\ index = 7$

6.3.3 Tests sur K-means

L'algorithme K-means est un algorithme très connu dans le domaine de la classification. Les tests effectués ont pour objet de montrer les résultats de segmentation obtenus en modifiant le nombre initial de classes de l'algorithme. Les figures 6.8 et 6.9 montrent les résultats obtenus pour chaque valeur du nombre initial de classes à savoir : 5, 10, et 20.

6.3.4 Tests sur Fuzzy-C-means

Le paramètre intéressant à tester dans l'algorithme Fuzzy-C-means est l'indice flou (fuzzy index). Les figures 6.10 et 6.11 montrent l'influence de ce paramètre sur les résultats de la segmentation pour chaque valeur de l'indice flou à savoir : 2, 5, et 7.

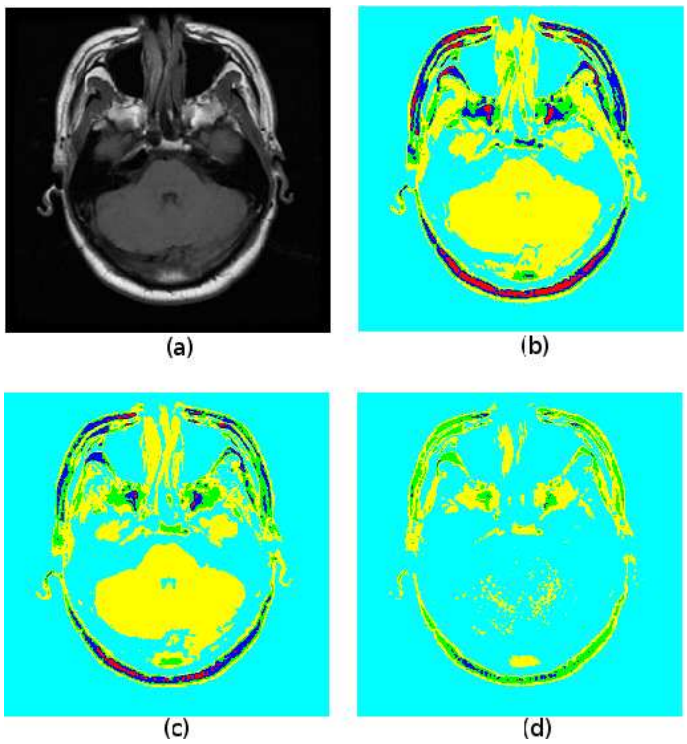

FIG. 6.11 – (a) Image initiale (Dhawan). (b) Segmentation par FCM avec $fuzzy\ index = 2$. (c) Segmentation avec $fuzzy\ index = 5$. (d) Segmentation avec $fuzzy\ index = 7$

6.4 Tests de performance

Le but de cette section est d'efectuer une étude comparative des différents algorithmes de segmentation implémentés. Cette comparaison porte les paramètres ci-aprés : nombre de classes, temps d'exécution, indices d'évaluation supervisés et non supervisés.

6.4.1 Méthodologie des tests

Nous avons implémenté deux types de méthodes pour évaluer les résultats d'une segmentation : les méthodes non supervisées, et la méthode supervisée.

Les méthodes d'évaluation non-supervisées estiment la qualité d'une segmentation en se basant sur l'image segmentée. Ces méthodes fournissent une estimation des distances qui séparent les différentes classes, ainsi que des distances entre les éléments d'une même classe. Nous avons utilisé plusieurs indices d'estimation existants. Parmi ces indices, on peut citer celui de [Turi, 2001] et [Amrane, 2004].

La méthode d'évaluation supervisée compare le résultat d'une segmentation avec une segmentation de référence réalisée par un spécialiste. Dans cette méthode, c'est

le degré de correspondance entre les régions de la segmentation résultante et les régions de la segmentation de référence qui détermine la qualité d'une segmentation. Nous utilisons pour cela, l'indice de *Jaccard* qui estime le degré de correspondance entre deux segmentations.

Dans le cadre de notre travail, nous avons choisi de nous concentrer sur les algorithmes suivants[1] :

AiNet : Avec les paramètres suivants :

Iterations	σ_s	σ_d	N	$\zeta\%$
1	35	35	3	0.3

Particle Swarm Optimization (PSO) : Avec les paramètres suivants :

Iterations	W	$C1$	$C2$	Swarm	Particle	Velocity
25	0.470	1.490	1.490	20	5	5

Fuzzy C-Means : Avec les paramètres suivants :

Iterations	Clusters	Fuzziness Index
100	5	2.0

K-means : Avec les paramètres suivants :

Iterations	Clusters
100	5

PSO + Fuzzy C-Means Avec les paramètres suivants, pour le PSO et le Fuzzy C-Means respectivement :

Iterations	W	$C1$	$C2$	Swarm	Particle	Velocity
25	0.470	1.490	1.490	20	5	5

Iterations	Clusters	Fuzziness Index
100	5	2.0

AiNet + Fuzzy C-Means : Avec les paramètres suivants pour AiNet et le Fuzzy C-Means respectivement :

Iterations	σ_s	σ_d	N	$\zeta\%$
1	35	35	3	0.3

Iterations	Clusters	Fuzziness Index
100	5	2.0

L'approche suivie est de tester les algorithmes ci dessus cités sur les deux benchmarks mentionnés précédemment (IBSR et Dhawan) puis de calculer les moyennes des indices d'évaluation obtenus pour chaque image. Spécifiquement, pour chaque benchmark, et pour chaque algorithme, les valeurs suivantes sont représentées :

– Moyenne du nombre de classes générées ;

[1]Les paramètres des différents algorithmes implémentés ont été choisis de façon empirique

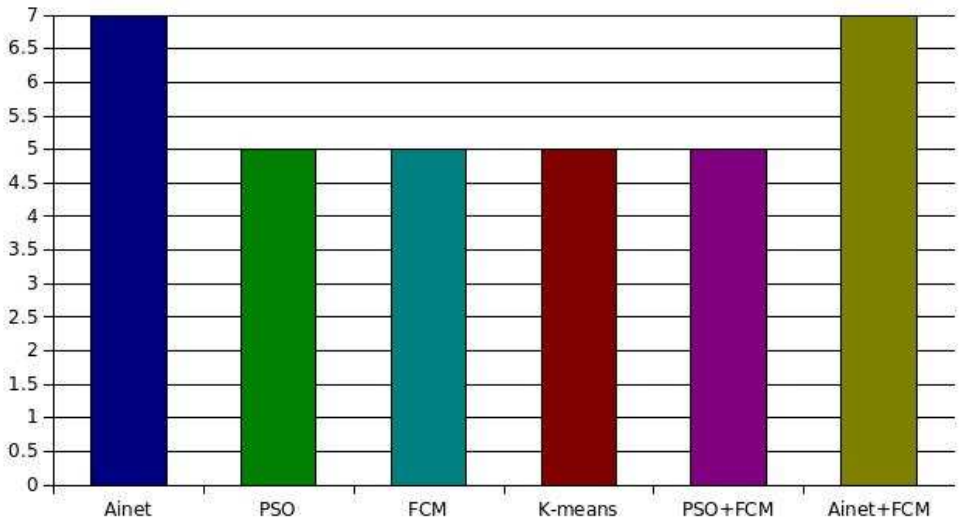

FIG. 6.12 – Moyennes du nombre de classes générées pour le benchmark IBSR

- Moyenne des temps d'exécution ;
- Moyenne des indices de Turi obtenus pour chaque image ;
- Moyenne des indices de Amrane obtenus pour chaque image ;

La moyenne des indices de Jaccard (critère d'évaluation supervisé) pour le benchmark IBSR est également représentée.

Les résultats obtenus sont représentés dans les figures 6.12, 6.13, 6.14, 6.15, et 6.16 pour le benchmark IBSR, et dans les figures 6.19, 6.20, 6.21 et 6.22 pour le benchmark Dhawan.

6.4.2 Analyse des résultats

Les paragraphes suivants présentent l'analyse qui ressort des résultats obtenus, après application des algorithmes de segmentation sur les deux benchmarks : IBSR et Dhawan. Nous présentons d'abord l'analyse des résultats obtenus sur le benchmark IBSR, puis nous effectuons une étude comparative avec les résultats obtenus sur le benchmark Dhawan.

La figure 6.12 montre le nombre de classes obtenues pour chaque algorithme de segmentation. Nous pouvons constater que les deux algorithmes : *Ainet* et *Ainet+PSO* produisent 7 classes, tandis que les autres algorithmes produisent 5 classes. Ceci est dû au fait que le nombre de classes dans *Ainet* est généré dynamiquement, alors que dans les autres algorithmes le nombre de classes est un paramètre d'initialisation (ici fixé à 5).

La figure 6.13 montre le temps moyen d'exécution pour chaque algorithme sur le benchmark *IBSR*. Ces temps d'exécution sont obtenus grâce à la représentation

Application des algorithmes de segmentation sur les deux benchmarks : IBSR et Dhawan

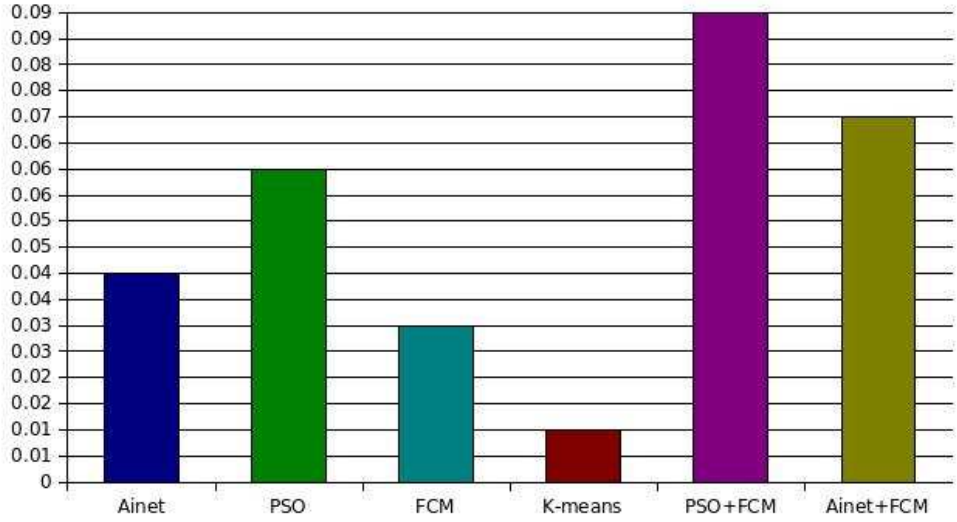

FIG. 6.13 – Moyennes des temps d'exécution (secondes) pour le benchmark IBSR

«abrégée» de l'image (voir la section 5.4). Nous remarquons que le *k-means* est plus rapide que les autres algorithmes, ceci est dû à sa simplicité par rapport à *Ainet* qui simule un comportement physiologique complexe, ou à *PSO* qui s'inspire du déplacement collectif d'animaux sociaux et qui nécessite plus de calculs.

Les figures 6.14, et 6.15 représentent des indices d'estimation non supervisée des résultats d'une segmentation. Comme nous l'avons déjà mentionné, le problème de la segmentation d'images est considéré comme un problème de partitionnement de données. Par conséquent, l'évaluation non supervisée représente l'évaluation des distances interclasses (qui doit être relativement grande), et des distances intraclasse (qui doit être relativement petite). Les différentes formules de Turi et de Amrane sont présentées respectivement dans les sections : 3.2.5.3, 5.5.1.1.

Les observations qui ressortent de l'analyse des graphes des indices non supervisés sont que l'algorithme *Ainet* est le meilleur d'un point de vue séparation et compacité selon l'indice de Turi. L'algorithme *PSO* est favorisé par l'indice d'estimation de Amrane du fait que le déplacement des particules est guidé par cet indice (voir la section 5.5.1.1). Enfin, sur le plan de la classification, les algorithmes implémentés donnent des résultats satisfaisants en comparaison avec des algorithmes classiques tels le *K-means* et le *Fuzzy-C-means*.

La figure 6.16 représente la moyenne des résultats de la comparaison entre le résultat de la segmentation et la segmentation de référence pour chaque algorithme sur le benchmark *IBSR*. Nous pouvons constater que le taux de correspondance entre les segmentations des IRM et leurs références ne dépasse pas, en moyenne, les 50% (figure 6.16).

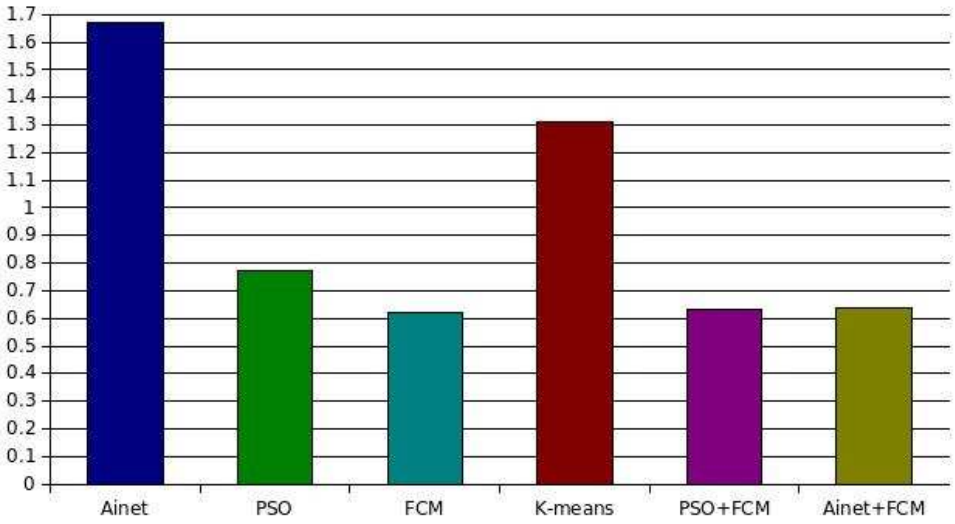

FIG. 6.14 – Moyennes des indices de Turi pour le benchmark IBSR

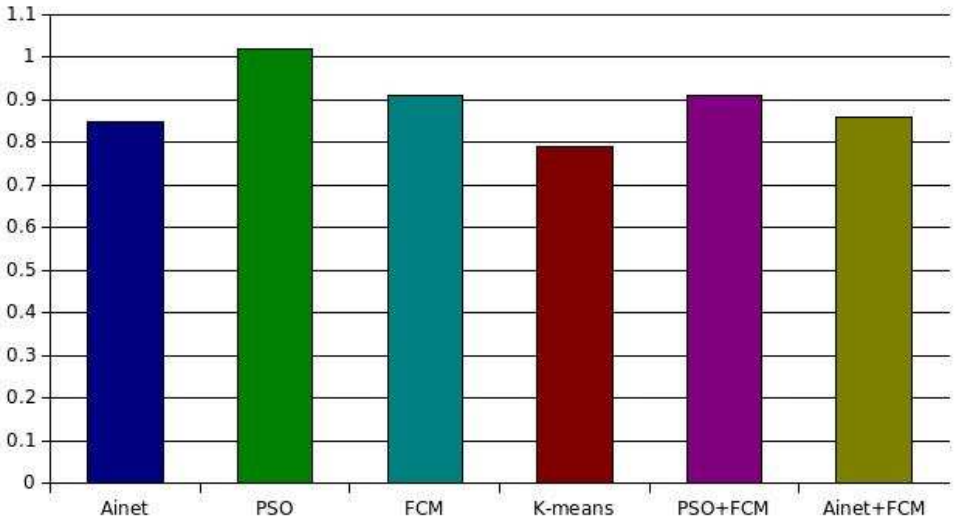

FIG. 6.15 – Moyennes des indices de Amrane pour le benchmark IBSR

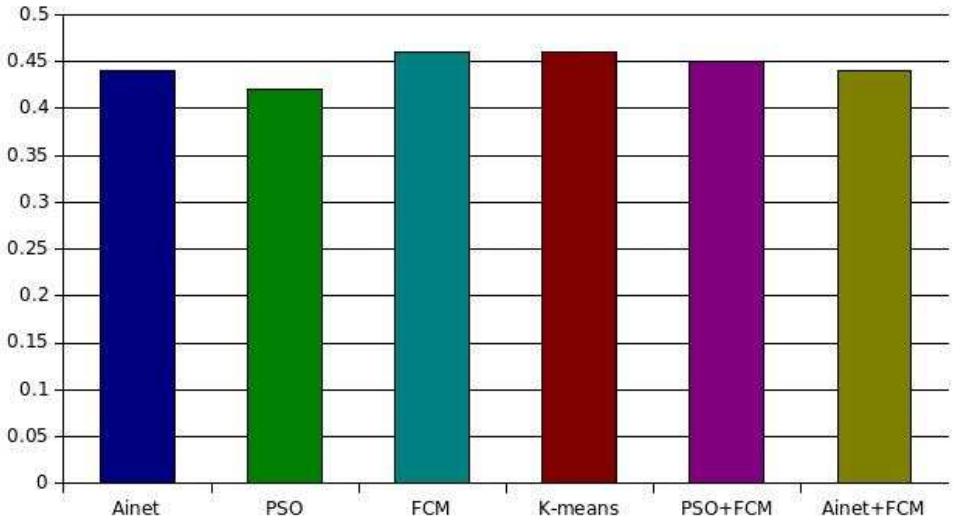

FIG. 6.16 – Moyennes des indices de Jaccard (Evaluation Supervisée) pour le benchmark IBSR

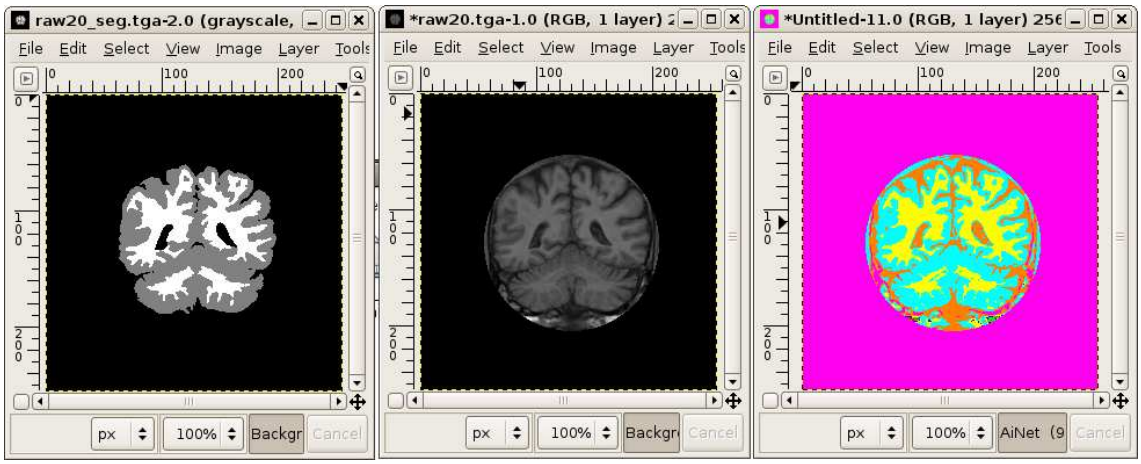

FIG. 6.17 – Partie d'une image IRM segmentée, avec sa segmentation de référence

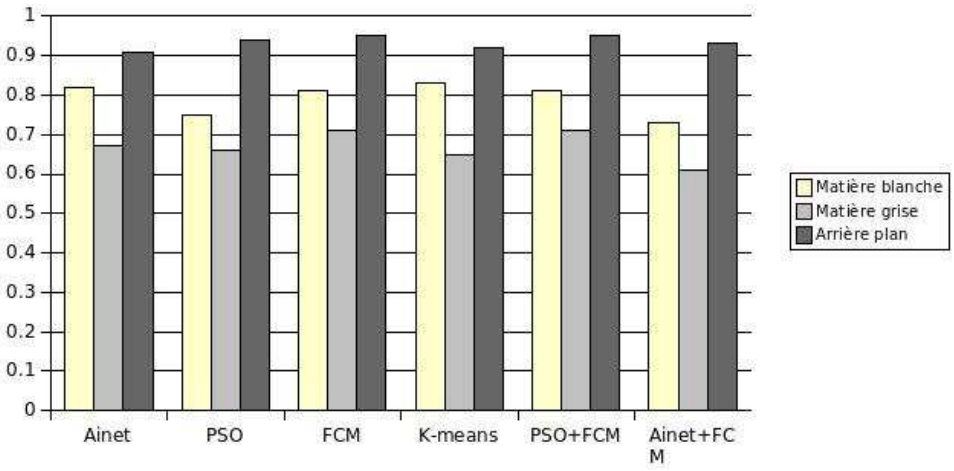

FIG. 6.18 – Comparaison des différents algorithmes de segmentation sur une partie d'une image IRM

Ce taux assez faible est dû au fait que les images supervisées du cerveau ne considèrent que la matière blanche et la matière grise dans une image. Or, les images IRM à segmenter sont «brutes», et représentent des coupes entières comprenant tout le système nerveux, en plus des os et d'autres structures anatomiques telles les oreilles dans certaines images. Pour avoir des résultats beaucoup plus fiables, l'utilisateur peut sélectionner la partie qui l'intéresse, en l'occurrence le cerveau, et appliquer les méthodes de segmentation choisies. La figure 6.17 montre les résultats de la segmentation d'une image IRM après élimination des parties non visées. La comparaison des différents algorithmes de segmentation est présentée dans la figure 6.18.

La figure 6.18 montre pour chaque algorithme de segmentation, le degré de correspondance des résultats obtenus (sur l'image du milieu dans la figure 6.17) avec la segmentation fournie par l'expert (image à droite dans la figure 6.17). L'axe des abscisses représente (comme dans tous les schémas qui suivent) les algorithmes testés, et l'axe des ordonnées représente le pourcentage de correspondance entre les structures anatomiques du cerveau (classes originales du benchmark supervisé), et les classes obtenues. Une valeur de 1 dans l'axe des ordonnées correspond à une identification parfaite entre la classe dans l'image segmentée et la structure anatomique correspondante dans la segmentation de référence.

Pour chaque algorithme dans l'axe des abscisses, la barre blanche représente le pourcentage de correspondance entre la classe désignant la matière blanche du cerveau dans le résultat de segmentation et la classe de la matière blanche attendue

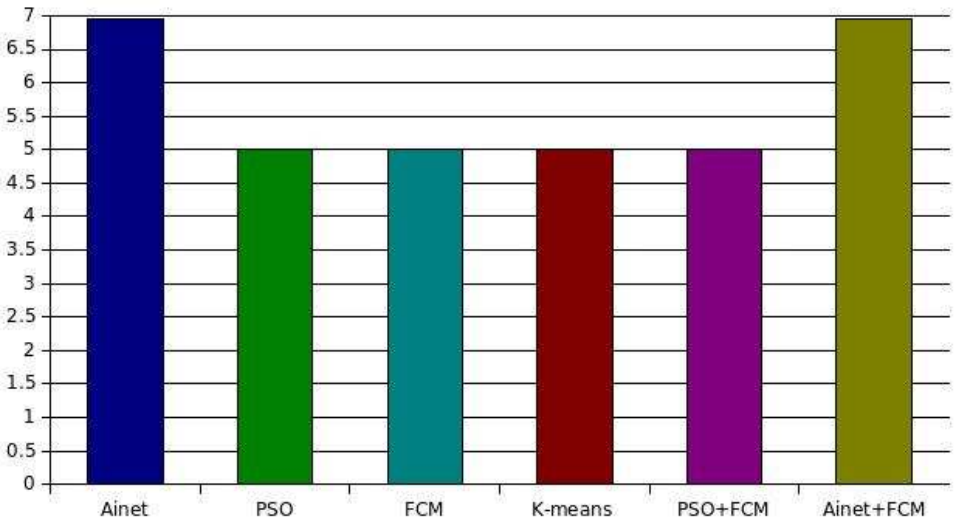

FIG. 6.19 – Moyennes du nombre de classes générées pour le benchmark Dhawan

par l'expert. La barre gris clair représente la matière grise du cerveau, et la barre gris foncé représente l'arrière plan.

Nous remarquons que tous les algorithmes donnent des résultats corrects, mais ce qui apparaît le plus, ceux sont les résultats que donne l'algorithme *Ainet*, qui, à la différence des autres, est indépendant d'une initialisation du nombre de classes. Nous remarquons aussi que les résultats du

PSO sont améliorés quand il est suivi du *FCM*, ceci nous permet d'avancer que d'autres hybridations peuvent améliorer les résultats d'une segmentation.

Les figures 6.19, 6.20, 6.21 et 6.22 représentent respectivement les moyennes du nombre de classes, du temps d'exécution, et des indices non supervisés pour chaque algorithme de segmentation sur le benchmark non supervisé de Dhawan [Dhawan, 2003].

Nous remarquons que les valeurs d'un même indice sur les deux benchmarks sont très similaires. La ressemblance des graphes des moyennes du nombre de classes (figures 6.12 et 6.19) ainsi que la ressemblance entre les graphes des temps d'exécution (figures 6.13 et 6.20) peut être attribuée au fait que les deux benchmarks sont de même nature (ils se composent d'IRM du cerveau, et donc comportent la même palette de couleurs) et se composent d'images de tailles similaires.

Comme l'indique les figures 6.13 et 6.20, le fait d'enchaîner des algorithmes les uns à la suite des autres affecte grandement la vitesse d'exécution de l'ensemble. Ainsi les algorithmes AiNet+FCM et PSO+FCM sont les plus lents. Ce point est à prendre en considération lorsqu'on envisage d'employer de telles hybridations sur de grandes quantités de données.

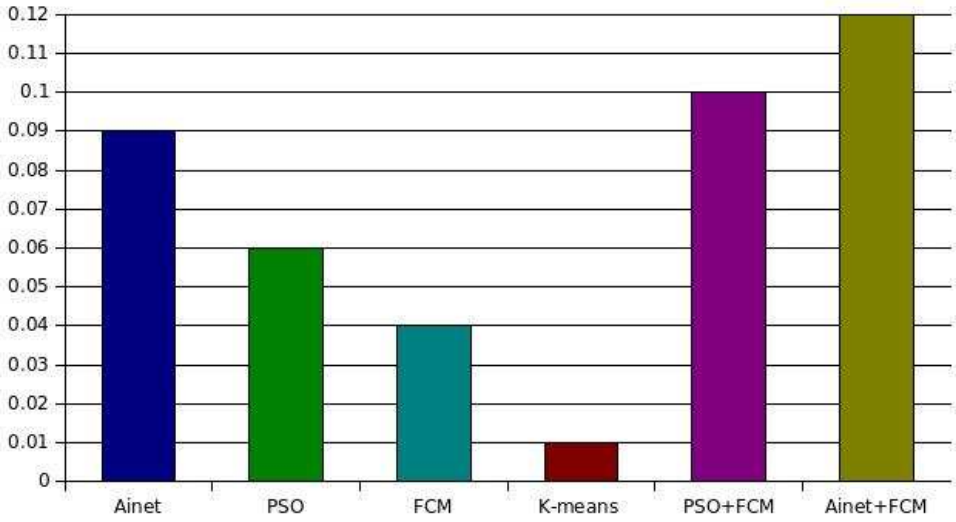

FIG. 6.20 – Moyennes des temps d'exécution (secondes) pour le benchmark Dhawan

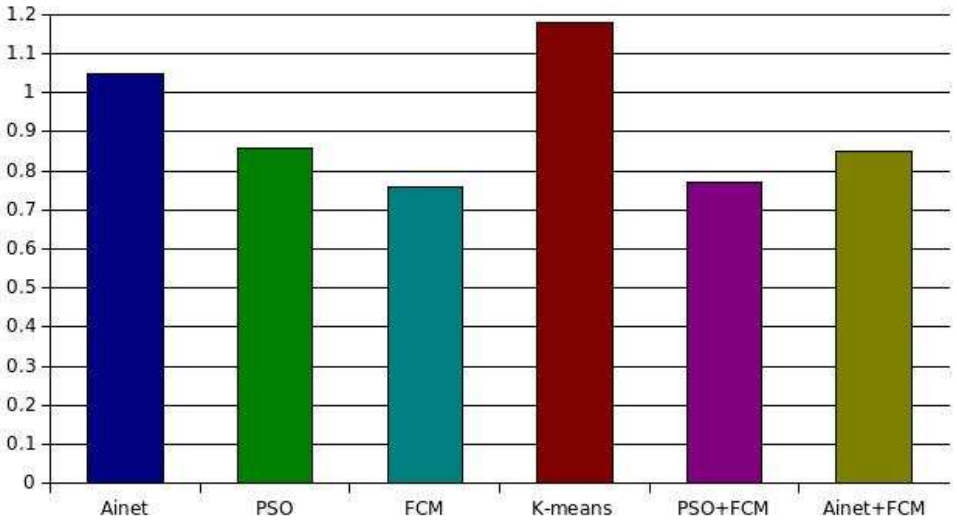

FIG. 6.21 – Moyennes des indices de Turi pour le benchmark Dhawan

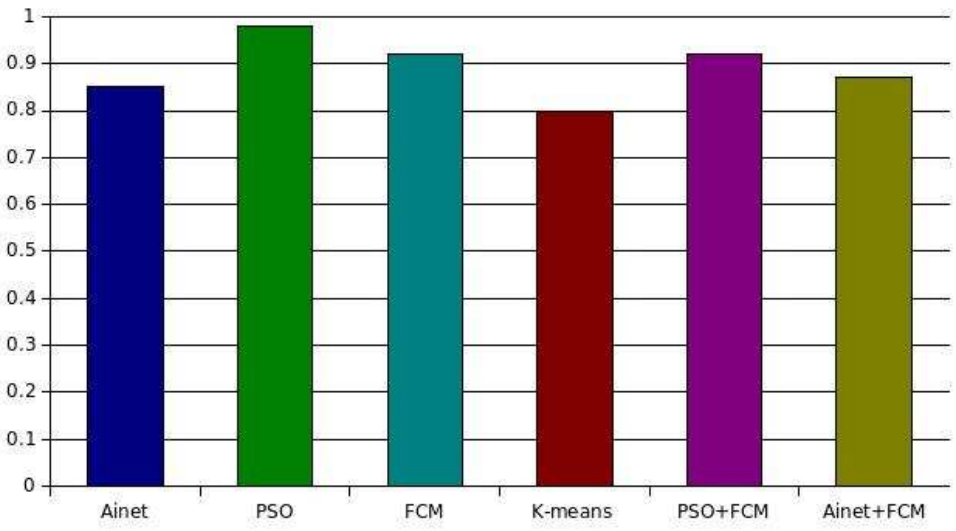

FIG. 6.22 – Moyennes des indices de Amrane pour le benchmark Dhawan

6.5 Conclusion

Dans ce chapitre, nous avons présenté une étude comparative des différents algorithmes implémentés dans notre application. Des indices d'estimation d'une classification ont été utilisés pour évaluer les résultats des algorithmes. Une estimation supervisée a été présentée dans le but d'étudier la fiabilité d'une segmentation automatique par rapport aux résultats attendus par des spécialistes.

Nous avons montré comment les résultats (dans le cas supervisé) d'une segmentation, peuvent être améliorés par une simple sélection de la zone d'intérêt dans une image.

Les observations qui ressortent des résultats obtenus peuvent être résumées dans les points suivants :

- Les valeurs d'un même indice sur les deux benchmarks (IBSR et Dhawan) sont similaires, du fait que les benchmarks sont de même nature.
- Le fait d'enchainer des algorithmes augmente considérablement le temps d'exécution.
- D'un autre côté, l'hybridation de PSO avec FCM améliore sensiblement les résultats.
- Ainet donne de bons résultats compte tenu du fait qu'il est indépendant d'une initialisation du nombre de classes.
- Le fait de représenter l'image par son histogramme réduit considérablement les temps d'exécution.
- L'algorithme PSO donne de bons résultats selon l'indice de Amrane, ce qui était prévisible, du fait que PSO optmise la fonction de Amrane dans le dé-

placement des particules.

Chapitre 7

Conclusion générale

Le travail présenté dans ce rapport a deux principaux objectifs : Etudier le comportement de deux algorithmes biomimétiques de partitionnement de données pour le problème de la segmentation d'images, et réaliser une plateforme extensible, pour pouvoir tester et hybrider des algorithmes de classification respectant un schéma spécifique.

Les algorithmes biomimétiques étudiés sont *Ainet* et *PSO*. Le premier, est un algorithme de partitionnement de données inspiré du mécanisme des réactions immunitaires des vertébrés. Le second, est un algorithme d'optimisation combinatoire inspiré du déplacement collectif observé chez les oiseaux migrateurs, dont le but est de trouver une solution *optimale* dans un ensemble (souvent très grand) de solutions candidates. Dans un but comparatif, nous avons aussi implementé des algorithmes *classiques* de classification tel le *K-means* et le *Fuzzy-C-Means*, afin de pouvoir les hybrider avec *Ainet* et *PSO* pour améliorer leurs résultats.

L'application réalisée est un greffon dans le logiciel d'édition d'images libre GIMP. L'intérêt est de tirer parti des fonctionnalités existantes et de construire l'application sur une plateforme évolutive et libre, assurant ainsi son utilisation et son amélioration ultérieure, dans un grand cadre communautaire.

Les images sur lesquelles les tests ont été effectués, sont des images médicales de type IRM du cerveau, dans le but d'automatiser la séparation des différentes structures anatomiques, et éventuellement, déceler des anomalies.

Les tests effectués sur les différents algorithmes ont montré l'intérêt de l'hybridation pour améliorer les résultats du *PSO*, ainsi que les résultats assez satisfaisants que donne Ainet pour la segmentation d'images.

7.1 Perspectives

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

1. Faire une étude plus approfondie pour le choix des paramètres de *Ainet* et du *PSO* pour affiner les résultats.
2. Ajouter des nouveaux indices d'estimations, selon le type d'images à segmenter, afin d'avoir une comparaison plus fiable des algorithmes.
3. Modéliser d'autres algorithmes biomimétiques et les comparer avec les résultats obtenus.
4. Tester les méthodes implémentées sur d'autres types d'images, comme les images satellitaires.

Annexe A

Concepts fondamentaux de l'imagerie médicale

A.1 Introduction

Les deux dernières décennies ont été marquées par des progrès majeurs dans le domaine de l'imagerie médicale et du traitement des images médicales assisté par ordinateur. L'importance clinique des techniques de d'imagerie radiologique dans le diagnostique et le traitement des maladies est devenue incontestable.

Les techniques de l'imagerie les plus couramment rependues sont :

- Les rayons X par tomographie axial informatisée (X-ray CT) ;
- l'imagerie par résonance magnétique (IRM) ;
- La tomographie informatisée par émission d'un photon unique (SPECT) ;
- La tomographie par émission de positrons (PET) ;
- L'imagerie par ultrasons.

Il est à noter que ces nouvelles méthodes d'imagerie médicale impliquent des équipements électroniques sophistiqués pour l'acquisition, la reconstruction et l'affichage des images.

Les méthodes d'imagerie radiologique planaire telles la mammographie (examen radiologique des seins et de la glande mammaire) fournissent elles aussi des images analogiques de haute qualité, représentant des projections bidimensionnelles des organes anatomiques tridimensionnels.

Par contre, les techniques récentes de l'imagerie tel les IRM, SPECT, PET, et l'ultrason dépendent fortement des technologies informatiques pour la création et l'affichage des images numérique. En utilisant des ordinateurs, les images numériques multidimensionnelles des structures physiologiques peuvent être traitées et manipulées pour la visualisions des formes cachées difficiles ou impossibles à voir sur

des image bidimensionnelles. De plus, l'analyse des formes cachées peut être faite à l'aide de programmes (informatiques) , et ce pour aider le praticien dans la prise de décisions. Malgré tout, les méthodes planaire (tel la mammographie) ne doivent pas être sous-estimées. En plus de la rentabilité et de la fiabilité des outils d'analyse, elles permettent d'obtenir des informations importantes, souvent suffisante pour un diagnostique.

A.2 Techniques d'imagerie médicale

De nombreuses techniques d'imagerie médicale ont été développées, elles sont généralement classées selon la manière d'interaction physique entre le sujet et l'appareil d'acquisition.

A.2.1 Imagerie par rayons X

Les rayons X ont été utilisés pour localiser des corps étrangers, tels que des balles, à l'intérieur du corps humain. Avec l'amélioration des techniques d'examen par rayons X, la radiographie a pu révélé d'infimes altérations des tissus et de nombreux états pathologiques ont pu être diagnostiqués par ce moyen.

Les rayons X ont fourni la plus importante méthode de diagnostic de la tuberculose lorsque cette maladie s'est déclarée. Les images des poumons étaient faciles à interpréter, car les espaces remplis d'air sont moins opaques aux rayons X que les tissus pulmonaires. Diverses autres cavités corporelles peuvent être remplies artificiellement avec des substances de contraste plus ou moins opaques que les tissus environnants, de manière qu'un organe particulier apparaisse plus distinctement sur l'image.

Un appareil de radiographie à rayons X offre des vues anatomiques claires de n'importe quelle partie du corps humain, y compris des tissus mous. L'appareil de tomographie axiale informatisée (CAT ou CT) tourne de 180° autour du sujet, en émettant un faisceau de rayons X. Des cristaux placés face au faisceau recueillent ce dernier et enregistrent les taux d'absorption des diverses épaisseurs de tissus et d'os. Ces données sont ensuite retransmises à un ordinateur, qui les transcrit dans une image à l'écran. Le scanner a été inventé dans les années 1970 par Godfrey Hounsfield, ingénieur en électronique britannique, et il devint d'usage courant vers 1979.

FIG. A.1 – Radiographie par rayons X d'une mâchoire déboîtée.

FIG. A.2 – Radiographie du crâne.

A.2.2 Imagerie par Résonance Magnétique (IRM)

L'imagerie par résonance magnétique est une technique d'imagerie médicale se fondant sur les principes de la résonance magnétique nucléaire.

C'est entre 1930 et 1940 que furent conduites les recherches fondamentales sur les interactions entre le noyau de l'atome et les champs magnétiques. En 1950, les principes physiques fondamentaux de la résonance magnétique étaient déjà bien compris. Pourtant, il restait encore trois conditions à remplir : disposer d'un ordinateur suffisamment rapide et puissant, réaliser un aimant stable à taille humaine associé à des appareils radio et enfin imaginer une utilisation médicale de ces techniques. Lauterbur, Damadian et Mansfield ont démontré la faisabilité de cette idée en utilisant les principes physiques de la résonance magnétique nucléaire. Les premières images réalisées grâce à cette technique furent publiées au début des années 1970. Les applications médicales se sont considérablement développées dans les laboratoires et les centres médicaux du monde entier entre 1983 et 1993.

L'imagerie médicale par la résonance magnétique a une multitude d'applications. Les experts s'accordent à dire que l'IRM est la méthode de diagnostic la plus puissante et la plus sensible disponible actuellement. Pour donner une idée de son importance, disons simplement qu'elle permet d'obtenir des images de n'importe quel organe, dans n'importe quelle coupe, et ce dans un délai relativement court.

Le principe de l'IRM met à profit la distribution aléatoire des protons qui possèdent des propriétés magnétiques. Le processus se fait en trois étapes. Dans un premier temps, l'IRM place le corps dans un champ magnétique très puissant (30 000 fois plus puissant que celui de la Terre) qui oriente tous les protons dans la même direction. Ensuite, les protons sont excités par des ondes radio, qui modifient leur orientation. Enfin, la stimulation est brutalement interrompue, et l'appareil recueille une onde dite de « résonance » par des antennes spécialement conçues. L'analyse informatique du signal transmis permet d'établir les images des organes internes en utilisant des méthodes similaires à celles qui ont été mises au point pour la radiographie aux rayons X ou les scanners.

L'IRM est utilisée pour diagnostiquer des atteintes du cerveau et du système nerveux central. Les examens par IRM ont une résolution anatomique comparable à celle des scanners, mais un meilleur contraste. Ils fournissent le même type d'informations que la tomographie par émission de positons, mais avec plus de détails anatomiques.

Par ailleurs, l'IRM est très supérieure aux images à rayons X, car elle a la capacité de distinguer les différences d'intensités entre les tissus mous normaux et pathologiques.

FIG. A.3 – L'appareil IRM

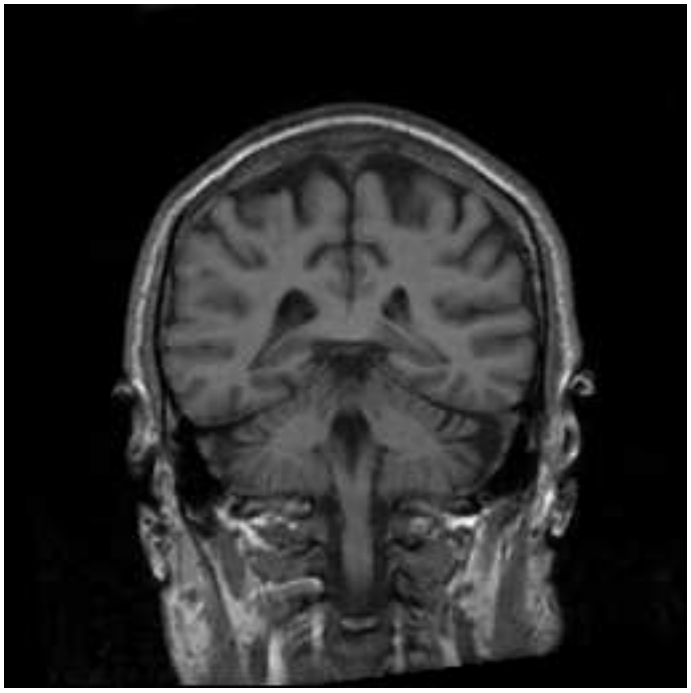

FIG. A.4 – IRM d'une coupe d'un encéphale.

A.2.3 Tomographie par émission de positrons (PET)

La tomographie par émission de positons (ou positrons) nécessite un cyclotron générateur de substances émettrices de positons (sortes de particules élémentaires), c'est-à-dire des isotopes (substances radioactives) à courte durée de vie.

L'isotope radioactif est fixé sur une substance que l'on veut étudier et que l'on injecte ensuite au patient. La substance radioactive (par exemple du glucose radioactif) va connaître le même destin que la substance naturelle (le glucose que l'on trouve normalement dans le corps) : elle va pénétrer de préférence dans certains organes, subir telle réaction chimique et donc disparaître à telle vitesse, etc. Mais, en outre, elle émet des positons, détectés par un compteur externe à scintillations, qui permettent de la repérer et de la suivre. Un ordinateur traite l'information et transmet des images et des données, qui concernent, par exemple, le flux sanguin et les processus métaboliques des tissus observés. L'image obtenue étant une vue en coupe de l'organe.

La tomographie par émission de positons est particulièrement utile pour étudier le cerveau et le cœur. Elle permet de diagnostiquer les tumeurs cérébrales et d'étudier les conséquences des troubles cardiaques sur les fonctions du cerveau, ainsi que divers troubles mentaux. Ces images sont également utilisées pour les recherches sur le cerveau et pour la cartographie des fonctions cérébrales. Cette technique est très coûteuse et très difficile à mettre en œuvre. Son intérêt diagnostique commence à être reconnu, mais il est difficile de prévoir si le passage à la pratique médicale se fera ou non. La tomographie par émission de positons est encore essentiellement un outil de recherche.

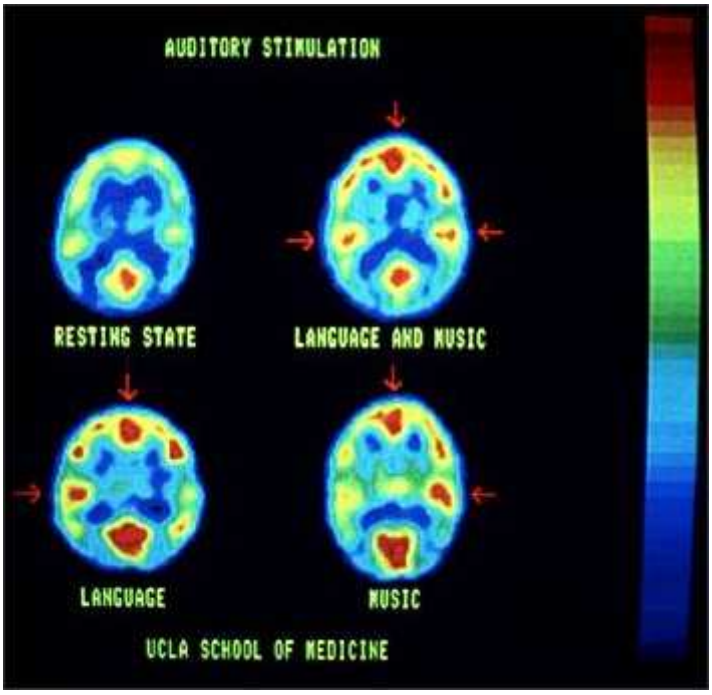

FIG. A.5 – Tomographie par émission de positrons image montrant les zones en activités du cerveau.

Bibliographie

[Ada and Nossal, 1987] Ada, G. L. and Nossal, G. (1987). The clonal selection theory. *Scientific American*, 2 :257.

[Amrane, 2004] Amrane, G. H. (2004). *Particle Swarm Optimization Methods for Pattern Recognition and Image Processing*. PhD thesis, University of Pretoria.

[Anderberg, 1973] Anderberg, M. (1973). *Cluster analysis for Applications*. Academic Press, New York.

[Azzag et al., 2003] Azzag, N., Monmarché, H., Slimane, M., Venturini, G., and Guinot, C. (2003). Anttree : a new model for clustering with artificial ants. *IEEE Congress on Evolutionary Computation, Canberra, Australia*.

[Baddeley, 1992] Baddeley, A. J. (1992). An error metric for binary images. *Robust Computer Vision*, pages 59–78.

[Ball and Hall, 1967] Ball, G. H. and Hall, D. J. (1967). A clustering technique for summarizing multi-variate data. *Behavioral Science*, 12.

[Bartels and Fisher, 1995] Bartels, K. A. and Fisher, J. L. (1995). Multifrequency eddy current image processing techniques for non-destructive evaluation. *International Conference on Image Processing*.

[Beauchemin et al., 1998] Beauchemin, M., Thomson, K. P. B., and Edwards, G. (1998). On the hausdorff distance used for the evaluation of segmentation results. *Canadian Journal of Remote Sensing (CJRS)*.

[Bezdeck, 1981] Bezdeck, J. C. (1981). *Pattern recognition with fuzzy objective function algorithms*. Plenum Press Ed, New York.

[Borsotti et al., 1998] Borsotti, M., Campadelli, P., and Schettini, R. (1998). Quantitative evaluation of color image segmentation results. *Pattern Recognition Letters*, 19 :741–747.

[Burnet, 1978] Burnet, F. M. (1978). Clonal selection and after.

[Canny, 1986] Canny, J. (1986). A computational approach to edge detection. *On Pattern Analysis and Machine Intelligence*, 8(6).

[Castro and Zuben, 1999] Castro, L. N. D. and Zuben, F. J. V. (1999). Artificial immune systems : Part i – basic theory and applications. Technical report, State University of Campinas, Campinas, Brazil.

[Castro and Zuben, 2001] Castro, L. N. D. and Zuben, F. J. V. (2001). Ainet : An artificial immune network for data analysis. In Sarker, R. A., Abbass, H. A., and Newton, C., editors, *Data Mining : A Heuristic Approach*, chapter 12. Idea Group Publishing.

[Clerc, 2002] Clerc, M. (2002). L'optimisation par essaim particulaire : principes, modèles et usage. *RSTI-TSI*, 21 :941–964.

[Cocquerez and Philipp, 1995] Cocquerez, J. and Philipp, S. (1995). Comparaison de méthodes de segmentation d'images.

[Colorni et al., 1991] Colorni, A., Dorigo, M., and Maniezzo, V. (1991). Distributed optimisation by ant colonies. *In Proceeding of ECAL-91.*

[Davis and Bouldin, 1979] Davis, D. and Bouldin, D. (1979). A cluster separation measure. *Pattern Analysis and Machine Intelligence*, 1(2).

[Deneubourg et al., 1990] Deneubourg, J., Aron, S., Goss, S., and Pasteels, J. (1990). The self-organizing exploratory pattern of the argentine ant. *Journal on insect Behavior*, 3 :159–168.

[Deneubourg et al., 1991] Deneubourg, J. L., Goss, S., Franks, N., Sendova-Franks, A., Detrain, C., and Chretien, L. (1991). The dynamic of collective sorting robot-like ants and ant-like robots. In Meyer, J. A. and Wilson, S. W., editors, *SAB 90 - 1st Conference On Simulation of Adaptive Behavior : From Animals to Animats.* MIT Press.

[Deriche, 1990] Deriche, R. (1990). Fast algorithms for low level vision. *Pattern Analysis and Machine Intelligence.*

[Developez.com, 2007] Developez.com (2007). *Les algorithmes génétiques.*

[Dhawan, 2003] Dhawan, A. (2003). *Medical Image Analysis.* Wiley-IEEE Press.

[Dorigo et al., 1991] Dorigo, M., Maniezzo, V., and Colorni, A. (1991). Positive feedback as a search strategy. Technical report, Technical Report 91-16 Politecnico di Milano, Italy.

[Forgey, 1965] Forgey, E. (1965). Cluster analysis of multivarariate data : efficiency versus interpretability of classification. *Biometrics.*

[Frigui and Krishnapuram, 1999] Frigui, H. and Krishnapuram, R. (1999). A robust competitive clustering algorithm with applications in computer vision. *Pattern Analysis and Machine Intelligence*, 21(5).

[Fu and Mui, 1981] Fu, K. and Mui, J. (1981). A survey on image segmentation. *Pattern Recognition*, 13(3).

[Goldberg, 1989] Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley.

[Gonzalez and Woods, 2002] Gonzalez, R. and Woods, R. (2002). *Digital image processing*. Prentice Hall, second edition.

[Halkidi et al., 2001] Halkidi, M., Batistakis, Y., and Vazirgiannis, M. (2001). On clustering validation techniques. In *Intelligent Information Systems Journal*. Kluwer Pulishers.

[Hamerly, 2003] Hamerly, G. (2003). *Learning Structure and Concepts in Data using Data Clustering*. PhD thesis, University of California, San Diego.

[Haralick and Shapiro, 1985] Haralick, R. and Shapiro, L. (1985). Image segmentation techniques. *Computer Vision, Graphics and Image Processing*, 12 :100–132.

[Holland, 1975] Holland, J. (1975). *Adaptation in natural and artificial systems*. MIT Press.

[Huet and Philipp, 1998] Huet, F. and Philipp, S. (1998). Fusion of images interpreted by a new fuzzy classifier. *Pattern Analysis and Applications*, 1.

[ibsr, 2007] ibsr (2007). The internet brain segmentation repository (ibsr). *http ://www.cma.mgh.harvard.edu/ibsr/*.

[Janeway et al., 2000] Janeway, C. A., Travers, P., Walport, M., and Capra, J. D. (2000). The immune system in health and disease. Technical report, Artes Médicas.

[Jerne, 1974] Jerne, N. (1974). Towards a network theory of the immune system. Technical report, Inst. Pasteur.

[Kennedy and Eberhart, 2001] Kennedy, J. and Eberhart, R. C. (2001). *Swarm intelligence*. Morgan Kaufmann.

[Kepler and Perelson, 1993] Kepler, T. B. and Perelson, A. S. (1993). Somatic hypermutation in b cells : An optimal control treatment. *Journal of Theoretical Biology*, 1(164) :37–64.

[Labroche, 2003] Labroche, N. (2003). *Modélisation par système de reconnaissance chimique des fourmis pour le problème de la classification non-supervisée : application à la mesure d'audience sur internet*. PhD thesis, Université de Tours.

[Levine and Nazif, 1985] Levine, M. D. and Nazif, A. M. (1985). Dynamic measurement of computer generated image segmentations. *IEEE Transaction on Pattern Analysis and Machine intelligence*, pages 155–164.

[Liu and Yang, 1994] Liu, J. and Yang, Y. H. (1994). Multiresolution color image segmentation. *PAMI*, pages 689–700.

[Marr and Hildreth, 1980] Marr, D. and Hildreth, H. (1980). Theory of edge detection. *Proceedings of the Royal Society of London B207*.

[Martin, 2002] Martin, D. R. (2002). *An empirical approach to grouping and segmentation*. PhD thesis, University of California, Berkeley, USA, Berkeley, USA.

[Monmarché, 2000] Monmarché, N. (2000). *algorithme de fourmis artificielles : application à la classification et à l'optimisation*. PhD thesis, Université de Tours.

[Ouadfel, 2006] Ouadfel, S. (2006). *Contributions à la Segmentation d'images basées sur la résolution collective par colonies de fourmis artificielles*. PhD thesis, Universté de Batna, Algérie.

[Perelsen and Oster, 1979] Perelsen, A. S. and Oster, G. F. (1979). Theoretical studies of clonal selection : Minimal antibody repertoire size and reliability of self-nonself discrimination. *Journal of Theoretical Biololgy*, 1(81) :645–670.

[Rosenberg, 1999] Rosenberg, C. (1999). *Mise en oeuvre d'un système adaptatif de segmentation d'images*. PhD thesis, Université de Rennes I.

[Sahoo et al., 1988] Sahoo, P. K., Soltani, S., C.Wong, A. K., and Chen, Y. C. (1988). A survey of thresholding techniques. *CVGIP*, 41 :233–260.

[Schwefel, 1981] Schwefel, H. (1981). *Numerical Optimisation of Computer Models*. Wiley.

[Sneath and Sokal, 1973] Sneath, P. and Sokal, R. (1973). Numerical taxonomy. *Freeman, London, UK*.

[Stuzle and Hoos, 1999] Stuzle, T. and Hoos, H. (1999). The max-min ant system and local search for combinatorial problems. In *Meta-heuristics : advances and trends in local search paradigms for Optimization by S. Voss, I.H. Osman and C. Roucairol*, pages 313–329. Kluwer Academic Publishers.

[Sutton and Barto, 1998] Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning an Introduction*. A Bradford Book.

[Tou, 1979] Tou, J. (1979). Dynoc- a dynamic optimal cluster seeking technique. *International Journal of Computer and Information sciences*, 8(6).

[Tou and Gonzalez, 1974] Tou, J. and Gonzalez, R. (1974). *Pattern Recognition Principles*. Addison-Wesley, Massachussets, USA, Massachussets, USA.

[Turi, 2001] Turi, R. (2001). *Clustering-Based Color Image Segmentation*. PhD thesis, Monash University, Australia.

[Vinet, 1991] Vinet, L. (1991). *Segmentation et mise en correspondance de régions de paires d'images stéréoscopiques*. PhD thesis, Université de Paris IX Dauphine.

[Yasnoff et al., 1977] Yasnoff, W. A., Mui, J. K., and Bacus, J. W. (1977). Error measures for scene segmentation. *Pattern Recognition*.

[Zhang, 1996] Zhang, Y. J. (1996). A survey on evaluation methods for image segmentation. *Pattern Recognition*.