



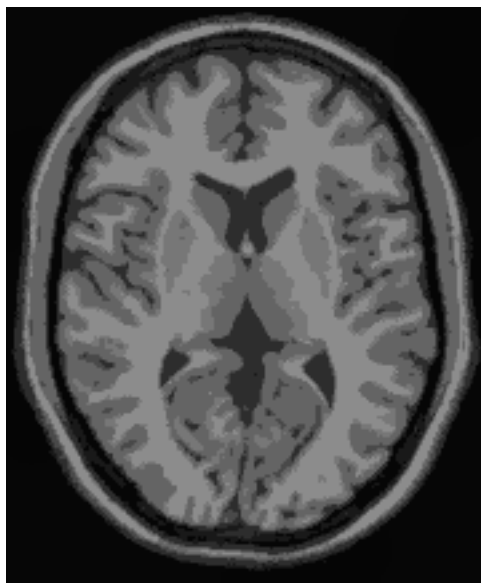
ENSPS
Parc d'innovation - Bd Sébastien Brant
67400 ILLKIRCH
Tél : 03 90 24 45 10
Fax : 03 90 24 45 45



Jean-Christophe Perles
ENSPS 3A - option ISPV
Master ISTI - spécialité IRMC

Mémoire de stage, diplôme d'ingénieur, Master M2 :
**Segmentation multiphase et multicanal
par contours déformables**

31 Août 2008



ENST/TSI
46 rue Barrault
75634 - PARIS Cedex 13
Maitre de Stage : Vincent Israel-Jost



REMERCIEMENTS

Mes premiers remerciements vont à Isabelle BLOCH pour m’avoir permis de réaliser mon stage au sein du laboratoire et pour m’avoir encouragé durant mon stage.

Tous mes remerciements à mon maitre de stage Vincent ISRAEL-JOST pour m’avoir accordé sa confiance et pour m’avoir laissé une grande liberté de travail. Grâce à lui, j’ai acquis une méthode de travail rigoureuse liée au travail de recherche pour ne pas être submergé par la quantité de données impressionnante qu’il faut traiter lorsqu’on travaille dans l’imagerie.

Je tiens à exprimer ma gratitude et mes remerciements à l’équipe “Traitement et Interprétation des Images”, en particulier à Elsa ANGELINI, Jérémie ANQUEZ, Geoffroy FOUQUIER, Hassan KHOTANLOU et Giampaolo FERRAIOLI pour leur aide, leur accueil ainsi que pour leur disponibilité.

Je remercie également les autres stagiaires (Benoit PETITPAS, Charles-Alban DELEDALLE, Yajing YAN, Renaud FALLOURD, Arpineh CHOLAKIAN, Moktar HORRI, Djinene LAYACHA, Rui WEI) pour leur aide occasionnelle et pour leur sympathie.

RÉSUMÉ

Le suivi médical des patients nécessite souvent l'utilisation d'imageurs médicaux tel qu'un IRM. La lecture et l'interprétation de ces images sont faites par le médecin pour diagnostiquer le patient. Pour l'aider, des techniques de traitement d'images sont mises en place et notamment la segmentation. Le but est de trouver les contours des parties anatomiques du cerveau (segmentation multiphase) et aussi d'utiliser conjointement les images d'une même coupe prises par deux imageurs différents (segmentation multicanal). Nous avons travaillé sur le modèle d'énergie de Mumford-Shah et sur des méthodes minimisant cette énergie.

Une première méthode de minimisation (Chan et Vese), utilisant les level-set et déjà implémenté au laboratoire, est trop lente et trop contraignante pour une application en milieu médical. Pour accroître les performances, nous avons développé deux autres méthodes. La première (le recuit simulé) permet d'atteindre le minimum global de l'énergie mais souffre d'un manque de rapidité. La deuxième (l'Iterated Conditional Mode) satisfait aux exigences d'efficacité et de rapidité d'exécution. En effet, en se basant sur des critères visuels et mathématiques (rapidité de calcul, comparaison avec des segmentations manuelles de tumeur, de matière blanche, de matière grise et de liquide céphalorachidien), nous avons montré que cette méthode était la meilleure parmi les trois testées.

ABSTRACT

In order to diagnose patients, doctors read and interpret images from the medical imaging as the MRI. To help them, image processing technologies, as segmentation, are set up. The aim is to find the outlines of brain anatomic parts (multi-phase segmentation) and also to use collectively the images of a same slice taken by two different protocols (multi-channel segmentation). We work on the energy model of Mumford-Shah and on energy minimisation methods.

A first level-set based minimisation method (Chan and Vese), which is already implemented in the laboratory, is too slow and too limiting for an application in medical field. To reach better performances, we have developed two other approaches. The first (the simulated annealing) obtains the global minimum of the energy but suffers of a lack of speed. The second (the Iterated Conditional Mode) satisfies the requirements of efficiency and execution speed. Indeed, based on visual and mathematic criteria (computation time, comparison with manual segmentation of tumors, white matter, grey matter and cerebrospinal fluid), we showed that this method was the best among the three tested.

Table des matières

Liste des abbréviations	8
Introduction	9
1 Présentation du cadre de travail	10
1.1 Présentation du laboratoire	10
1.2 Présentation du sujet de stage	10
2 Etat de l'art	12
2.1 Anatomie et images du cerveau	12
2.1.1 Anatomie du cerveau	12
2.1.2 Les images traitées et leurs caractéristiques	13
2.2 Les contours actifs sans bords	15
2.2.1 Le modèle de Mumford et Shah	15
2.2.2 La segmentation multiphase	17
2.2.3 La segmentation multicanal : les images vectorisées	20
2.2.4 La segmentation multicanal : le modèle logique	21
3 Algorithmes de segmentation multiphase et multicanal par minimisation de l'énergie de Mumford-Shah	24
3.1 Méthode de Chan et Vese	24
3.1.1 Segmentation multiphase et permutations	24
3.1.2 Fusion des données et segmentation multicanal	25
3.1.3 Résultats	26
3.2 Méthode du recuit simulé	32
3.2.1 Principe théorique	32
3.2.2 Pratique	33
3.2.3 La régularisation	34
3.2.4 Résultats	36
3.3 Méthode ICM	38
3.3.1 Principe de l'algorithme	38
3.3.2 Résultats	38
4 Initialisation automatique des phases	40
4.1 Initialisation des phases en segmentation monocanal	40
4.1.1 Première méthode	40
4.1.2 Deuxième méthode	41
4.2 Initialisation des phases en multicanal	42
4.2.1 Première méthode	42
4.2.2 Deuxième méthode	42
5 Application de la segmentation sur des images réelles	46
5.1 Critères de comparaison	46
5.2 Segmentation d'un cerveau en 3 phases : matière grise, matière blanche et LCR	47
5.2.1 Prétraitement des images	47

5.2.2	Résultat bruts	48
5.2.3	Post-traitements	48
5.3	Segmentation des tumeurs	49
5.3.1	Images traitées	49
5.3.2	Segmentation de l'image SPGR	50
5.3.3	Segmentation de l'image FLAIR	50
5.3.4	Segmentation multicanal des images SPGR-FLAIR	51
5.4	Segmentation d'un cerveau entier avec une tumeur	52
Conclusion		55
Bibliographie		56
Annexes		57
A	Relation mathématique entre les indices de similarité et de Jac- card	57
B	Etudes paramétriques de la segmentation sur IBSR	58
C	Etude paramétrique de la segmentation de la tumeur du cas 1 en SPGR	59
D	Etude paramétrique de la segmentation de la tumeur du cas 1 en FLAIR	60
E	Etude paramétrique de la segmentation multicanal de la tumeur du cas 1	61
F	Manuel d'utilisation des codes Matlab	62
G	Codes Matlab pour l'ICM	64
H	Interface graphique pour segmenter des images par l'ICM	67

Table des figures

1	Anatomie du cerveau. (extrait du site fr.brainexplorer.org)	12
2	Images extraites des données Brainweb	13
3	Images extraites des données d'IBSR	14
4	Images extraites de deux cas atteints d'une tumeur	14
5	Courbe $C=\{(x, y), \phi(x, y) = 0\}$	16
6	Segmentation d'un objet topologiquement complexe.[2]	17
7	Les deux fonctions level-set partitionnent l'image en 4 régions : $\{\phi_1 > 0, \phi_2 > 0\}$, $\{\phi_1 > 0, \phi_2 < 0\}$, $\{\phi_1 < 0, \phi_2 > 0\}$, $\{\phi_1 < 0, \phi_2 < 0\}$	18
8	Evolution de la segmentation d'une image IRM de cerveau par la méthode de Chan et Vese en utilisant deux fonctions level-set (à gauche : initialisation. à droite : résultat final). Les images de la première ligne représentent les contours des fonctions level-set et les images de la deuxième ligne représente les régions obtenues. [14] . .	19
9	Gauche : les lignes de niveaux $x \in \Omega : \phi(x) = 0$, $x \in \Omega : \phi(x) = 10$ et $x \in \Omega : \phi(x) = 20$ divisent l'image en quatre phases disjointes. Droite : les lignes de niveaux $x \in \Omega : \phi_i(x) = 0$ et $x \in \Omega : \phi_i(x) = 10$ $i = 1, 2$ divisent l'image en 9 régions disjointes.[4]	19
10	Il manque un bout du triangle sur chaque canal mais le contour final retrouve la totalité du triangle. [3]	21
11	Détection d'anomalies fictives dans le cerveau. On peut trouver les anomalies du canal 1 qui ne sont pas dans le canal 2 et inversement.[12]	23
12	Diagramme d'évolution des phases à 2 et 3 level-set	24
13	Première colonne (du haut vers le bas) : images du canal 1 et canal 2 formant trois 4 régions distinctes (dernière image) dont les corres- pondances des couleurs sont, blanc=100, noir=0, gris=10. Deuxième colonne : choix de trois initialisations différentes aboutissant au même résultat de segmentation (troisième colonne)	26
14	Segmentation de la coupe 80 de Brainweb avec différents coefficients de régularisation.	27
15	Segmentation de la coupe 80 de Brainweb avec un nombre différent d'itérations entre les permutations	28
16	Effet de la réinitialisation sur les level-set.	29
17	Segmentation de la coupe 80 de Brainweb avec réinitialisation après chaque permutation (à gauche) et à chaque itération (à droite) . . .	29
18	Segmentation de la coupe 80 de Brainweb avec un nombre différent de cycles de permutations. ($\mu = 0.1$, $S = 3$ et on a 20 itérations après la dernière permutation)	30
19	Calcul de l'énergie de régularisation (c). Plus un pixel est entouré de phases différentes de la sienne, plus l'énergie de régularisation sera élevée (blanc). A l'inverse si un point est uniquement entouré de pixels de sa phase alors l'énergie est nulle (noir).	35
20	Segmentation de la coupe 80 de Brainweb par l'algorithme du re- cuit simulé avec plusieurs paramètres de régularisation [autres pa- ramètres : 8 phases, $\alpha = 0.99$, 5000 itérations, $T^0 = 0, 1$].	37

21	Segmentation par ICM de la coupe 80 de Brainweb selon plusieurs paramètres de régularisation	39
22	Première ligne : carte de l'initialisation des phases selon les 3 méthodes (bornes inférieures (a), milieu des intervalles (b) et bornes supérieures (c)). Deuxième ligne : résultat de la segmentation avec les initialisations faites au dessus.	41
23	Initialisation des phases à partir de l'histogramme du canal 1. A gauche : initialisation du canal 1. A droite : initialisation du canal 2.	43
24	Schéma décrivant la méthode 2 d'initialisation multicanal. On rappelle que C_{ij} sont les valeurs moyennes des phases du canal i calculées à partir de l'histogramme du canal j . Supposons qu'on ait C11, C21 et C22. Pour mettre en correspondance les valeurs moyennes des phases du canal 1 (C11) et du canal 2 (C22), on utilise la transformation qu'il faut effectuer pour trier C21 dans l'ordre croissant. .	44
25	L'image en haut à gauche est P_{init1} et celle en haut à droite est P_{init2} . La mise en correspondance des phases s'effectue au moyen de C21 et on recalcule les labels des phases (en bas à gauche) comme le montre le changement de couleurs. On fusionne ensuite les deux images de gauche ce qui donne la carte d'initialisation finale (en bas à droite). (coupe 80 de Brainweb)	45
26	Schéma représentant les vrais positifs, faux positifs et faux négatifs	46
27	Segmentation de deux coupes de IBSR 5 en 4 phases : le fond (noir), la matière grise (orange), la matière blanche (blanc) et le LCR (bleu clair)	48
28	Segmentation en 4 phases après post-traitement (a) et (b) de deux coupes de IBSR à comparer avec les segmentations manuelles (c) et (d)	50
29	Evaluation des vrais positifs, faux positifs, indices de similarité et de Jaccard (en pourcentage) pour deux cas de tumeurs sur des segmentations monocanal (Mo) et multicanal (Mu) comparé aux segmentations manuelles associées.	51
30	Contours des tumeurs par segmentation ICM (rouge) et manuelle (en bleu) sur SPGR.	51
31	Contours des tumeurs par segmentation ICM (rouge) et manuelle (en vert) sur FLAIR	52
32	Contours des tumeurs par segmentation multicanal (rouge) et manuelle (vert pour FLAIR et bleu pour SPGR).	52
33	Segmentation avec un poids plus fort pour le canal SPGR	53
34	Courbes des vrais positifs de la matière blanche (bleu), de la matière grise (rose) et du LCR (jaune) en fonction du paramètre de régularisation.	58
35	Courbes des pourcentages de faux positifs de la matière blanche (bleu), de la matière grise (rose) et du LCR (jaune) en fonction du paramètre de régularisation.	58
36	Courbes des pourcentages de vrais (bleu) et faux (rose) positifs de la segmentation de la tumeur sur le canal SPGR en fonction du paramètre de régularisation.	59

37	Courbes des indices de similarité (bleu) et de Jaccard (rose) de la segmentation ICM de la tumeur sur l'image SPGR du cas 1 en fonction du paramètre de régularisation.	59
38	Courbes des pourcentages de vrais (bleu) et faux (rose) positifs de la segmentation de la tumeur sur le canal FLAIR en fonction du paramètre de régularisation.	60
39	Courbes des indices de similarité (bleu) et de Jaccard (rose) de la segmentation ICM de la tumeur sur l'image FLAIR du cas 1 en fonction du paramètre de régularisation.	60
40	Courbes des pourcentages de vrais (bleu) et faux (rose) positifs de la segmentation multicanal de la tumeur en fonction du paramètre de régularisation.	61
41	Courbes des indices de similarité (bleu) et de Jaccard (rose) de la segmentation ICM multicanal de la tumeur en fonction du paramètre de régularisation.	61
42	Cette interface permet d'utiliser simplement l'algorithme. On peut choisir l'image à segmenter, le répertoire de destination pour sauvegarder les résultats. L'interface permet de définir tous les paramètres et notamment de visualiser l'initialisation des phases avant de lancer l'algorithme. Sur le graphique de la fenêtre principale s'affiche l'évolution de l'énergie au cours des calculs. Enfin on peut visualiser les images segmentées et les phases obtenues dans le menu <i>View</i> . . .	67

Liste des abbréviations

IBSR : Internet Brain Repository

ICM : Iterated Conditional Mode

IRM : Imagerie par résonance magnétique

LCR : liquide céphalorachidien

Introduction

La segmentation a pour but de trouver automatiquement ou semi-automatiquement des objets dans une image. Cette tâche est en effet coûteuse à réaliser manuellement, notamment sur des images tridimensionnelles. Elle est pourtant nécessaire pour exploiter les images, en particulier dans le contexte médical où le suivi de l'évolution des pathologies repose souvent sur des mesures de volume d'organes, de tumeurs ou de lésions.

Dans ce travail, nous nous intéressons plus particulièrement à la possibilité de segmenter plusieurs objets simultanément sur une image (segmentation multi-région ou multiphase) ainsi qu'à l'exploitation de plusieurs images d'une même scène, acquises avec des imageurs ou des protocoles différents, pour réaliser la segmentation (segmentation vectorielle ou multicanal. En effet les images que nous étudierons sont des images d'IRM cérébrales à plusieurs niveaux de gris d'où l'intérêt de la segmentation multiphase. De plus, l'IRM permet d'acquérir plusieurs images pour une même coupe de cerveau, chacune mettant en évidence des structures différentes ce qui nécessite une segmentation multicanal.

Durant ce stage, les méthodes que j'ai étudiées appartiennent aux méthodes de segmentation par régions et sont basées sur la minimisation d'une fonction énergie donnée par Mumford et Shah. Cette énergie a la particularité d'être minimale lorsque les niveaux de gris des régions sont homogènes. La résolution de ce problème de minimisation peut se faire de plusieurs manières et nous allons en étudier trois dans ce rapport. La première a été donnée par Chan et Vese et utilise une formulation en level-set, la deuxième est le recuit simulé et la troisième est l'Iterated Conditional Mode (ICM). Ces méthodes doivent permettre de segmenter en multiphase et en multicanal.

Dans un premier temps, j'ai réalisé une étude bibliographique sur les méthodes classiques de segmentation par contours actifs et les méthodes par level-set ainsi que sur plusieurs modèles d'énergie avec en particulier celle de Mumford-Shah. Une deuxième partie de mon stage consistait à maîtriser les paramètres de la méthode déjà implémentée au laboratoire pour optimiser le processus de segmentation. Une troisième partie était l'implémentation d'autres algorithmes de minimisation de la même énergie et de comparer ces méthodes entre elles que ce soit en monocanal ou en vectoriel. Pour optimiser cette minimisation, une quatrième partie sera consacrée aux méthodes d'initialisation des algorithmes. Enfin je devais réaliser de multiples applications de ces algorithmes sur des images tests, des cerveaux simulés, ainsi que sur de vrais cerveaux, normaux et pathologiques.

1 Présentation du cadre de travail

1.1 Présentation du laboratoire

Mon stage s’est déroulé au sein du département TSI (Traitement du Signal et de l’Image) du groupe TELECOM ParisTech qui a pour mission l’enseignement, la recherche et la formation par la recherche dans les domaines du signal, des images et de l’application du traitement du signal.

Il existe 5 grandes structures dans ce département dont voici les thèmes abordés par chacun.

Le groupe “Traitements Statistiques et Applications aux Communications” travaille dans le domaine du signal pour les communications, la séparation de sources et sur des modèles statistiques pour le signal et l’image comme la restauration d’images.

Le groupe “Perception, Apprentissage et Modélisation” étudie le rôle des facteurs humains dans l’accès aux divers types d’information comme la parole, l’écrit, l’image et les interfaces multimodales.

Le groupe “Codage” travaille sur la compression des données et leur application dans le domaine de l’audiovisuel et du multimédia.

Le groupe “Audio, Acoustique et Ondes” étudie la physique des ondes dans le domaine de l’optique et de l’acoustique.

Le groupe “Traitement et Interprétation des Images (TII)” conduit des recherches dans l’analyse et l’interprétation d’images de scènes complexes. Les domaines d’application sont variés, cela va de l’imagerie médicale (imagerie cérébrale) à l’imagerie aérienne (imagerie radar et imagerie aérienne à très haute résolution des milieux urbains) en passant par la description des objets complexes tridimensionnels. C’est dans ce groupe composé de 17 chercheurs permanents et 45 doctorants que j’ai effectué mon stage.

1.2 Présentation du sujet de stage

Mon stage s’inscrit dans le projet de Vincent Israel-Jost, mon encadrant, qui réalise son post-doc au laboratoire. Il a implémenté une extension d’un algorithme de segmentation multiphase basé sur la minimisation de l’énergie de Mumford-Shah (voir partie 2.2.1 page 15) déjà implémenté au laboratoire en le rendant utilisable pour une segmentation multicanal. La segmentation multiphase consiste à segmenter une image en différentes régions, de manière à obtenir les contours des objets pertinents. La segmentation multicanal permet de fusionner les données de plusieurs images afin d’obtenir une meilleure segmentation. Mon travail consistait tout d’abord à comprendre et à utiliser l’algorithme déjà développé afin de dégager

des paramètres optimaux. Cet algorithme étant coûteux en temps et complexe à mettre en oeuvre, nous avons décidé ensuite d'implémenter deux autres algorithmes d'optimisation, l'un permettant de trouver théoriquement le minimum global de la fonction (le recuit simulé) et un autre permettant de trouver un minimum local (ICM). La comparaison de ces méthodes se fait sur divers critères tels que l'énergie finale, la qualité visuelle de la segmentation, ou encore la comparaison avec des segmentation manuelles. Parallèlement, nous avons étudié plus en détail la segmentation multicanal avec notamment le choix d'une règle de fusion des données de deux images.

2 Etat de l'art

2.1 Anatomie et images du cerveau

2.1.1 Anatomie du cerveau

Durant mon stage, j'ai appliqué mes algorithmes sur des images d'IRM cérébrales. Nous allons tout d'abord décrire brièvement les différentes structures de l'encéphale.

L'encéphale constitue notre système nerveux central avec la moëlle épinière. Il est composé de trois structures principales, le cerveau (ou prosencéphale), le cervelet (ou rhombencéphale) et le tronc cérébral (figure 1). Le cerveau est constitué de deux hémisphères qui eux mêmes contiennent 4 lobes (frontal, pariétal, occipital et temporal). La couche extérieure du cerveau est la matière grise et l'intérieur est la matière blanche. Dans cette dernière se trouvent aussi les noyaux gris centraux qui sont composés des noyaux caudés (caudate), du putamen et du pallidum (globus pallidus). Enfin les ventricules sont le siège de la sécrétion du liquide céphalorachidien (LCR).

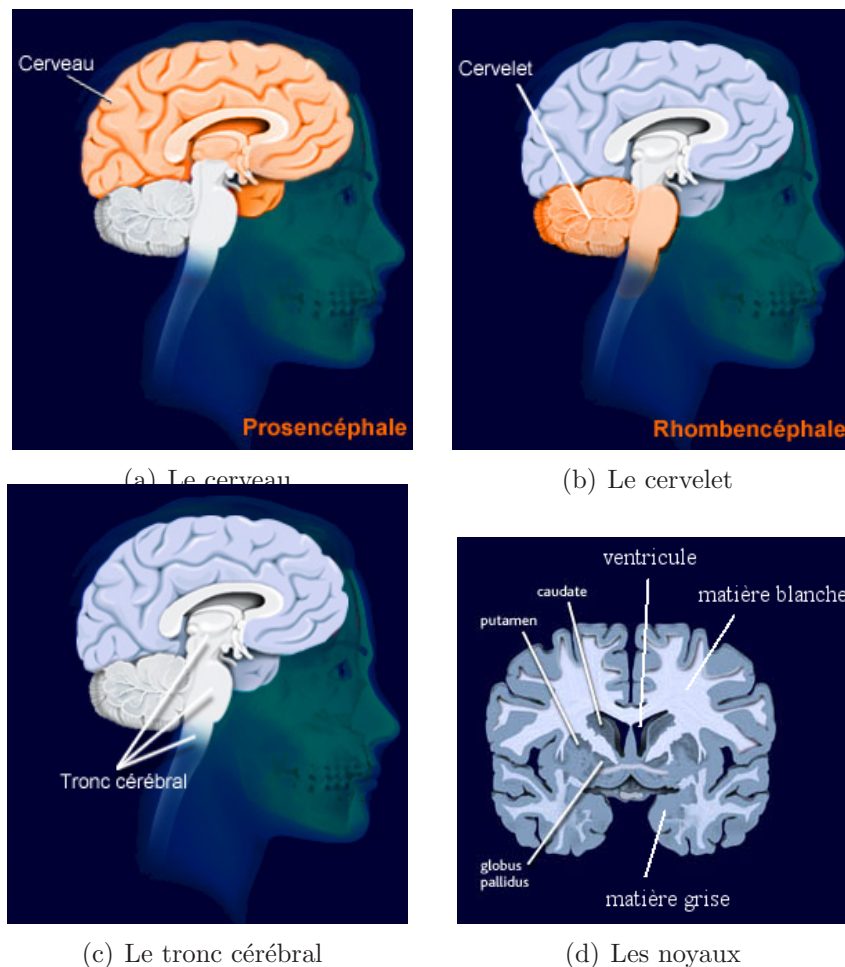


FIG. 1 – Anatomie du cerveau. (extrait du site fr.brainexplorer.org)

2.1.2 Les images traitées et leurs caractéristiques

Les images utilisées lors de mon stage sont de trois types. La première série d'images (Figure 2) appelée Brainweb provient du Centre d'Imagerie Cérébrale McConnel à l'Institut Neurologique de Montréal. Ce sont des images simulées du cerveau et en aucun cas ce ne sont de vraies images acquises. Nous avons notamment accès aux simulations d'acquisition IRM en mode T1 et T2 d'un cerveau virtuel. Le but de ces images est de créer un standard permettant d'évaluer les performances des algorithmes de traitement d'images. Nous pouvons aussi bien obtenir des images d'un cerveau sain que d'un cerveau atteint de lésions. Je n'ai cependant utilisé que les images d'un cerveau sain pour ce type d'image.

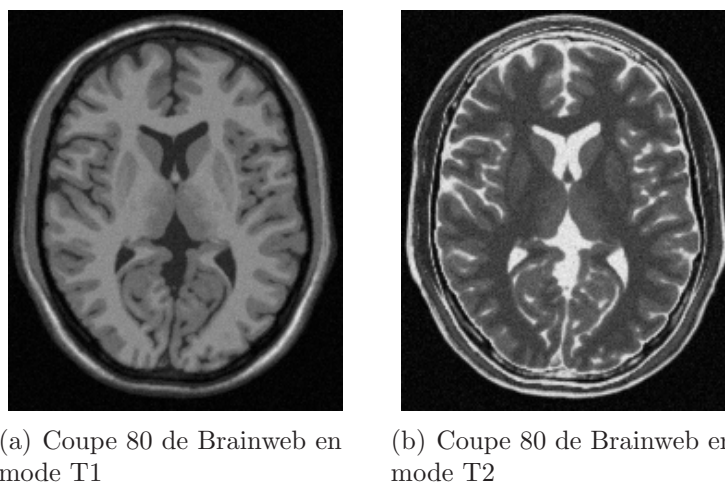
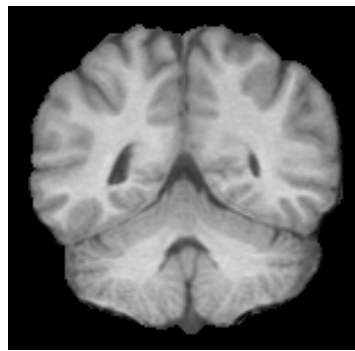


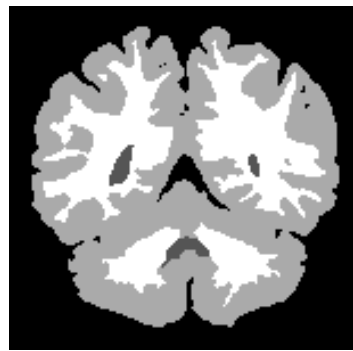
FIG. 2 – Images extraites des données Brainweb

La deuxième série d'images (figure 3) provient du Centre d'Analyse Morphométrique du Département de Neurologie de l'Hopital Général de Massachusetts. Cette banque de données appelée Internet Brain Segmentation Repository (IBSR) regroupe des vraies acquisitions IRM cérébrales avec les images des segmentations manuelles (matière blanche, matière grise et LCR) effectuées par des médecins. Ces images ont pour but d'évaluer les méthodes de segmentation automatique en les comparant avec les segmentations manuelles associées.

La troisième série d'images (figure 4) provient d'une banque de données de TSI. Ce sont 2 cas d'images d'IRM cérébrales réelles en mode SPGR et en mode FLAIR dont le cerveau est atteint par une tumeur. Ces deux modes mettent en valeur des structures différentes. En effet, le mode SPGR permet de bien visualiser les structures saines du cerveau (matière blanche, matière grise, LCR, noyaux) alors que le mode FLAIR offre un excellent contraste entre les zones cancéreuses (en clair sur la figure 4(b) et 4(d)) et les zones saines. On voit immédiatement l'intérêt d'une segmentation multicanal permettant à la fois de segmenter les parties saines visible en SPGR et aussi les parties malades visibles sur le FLAIR.

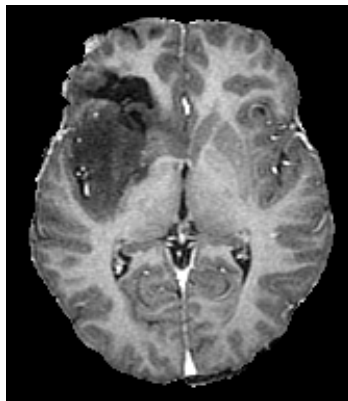


(a) Coupe 39

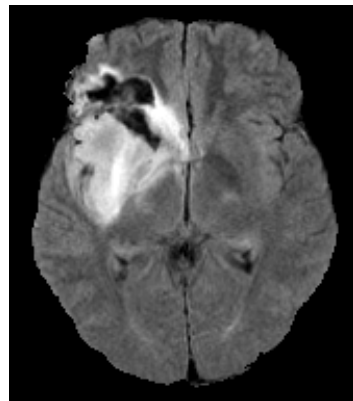


(b) Segmentation manuelle de la coupe 39

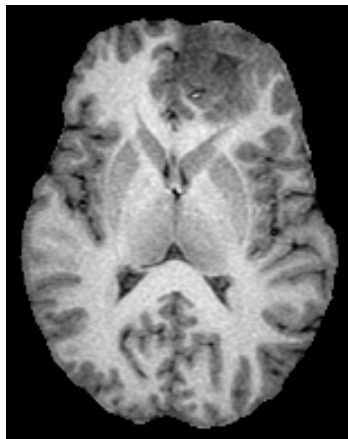
FIG. 3 – Images extraites des données d'IBSR



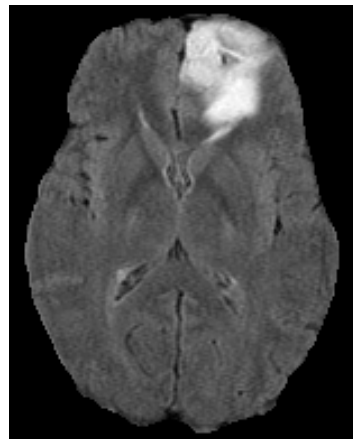
(a) Coupe sagittale du cas 1 en mode SPGR



(b) Coupe sagittale du cas 1 en mode FLAIR



(c) Coupe sagittale du cas 2 en mode SPGR



(d) Coupe sagittale du cas 2 en mode FLAIR

FIG. 4 – Images extraites de deux cas atteints d'une tumeur

2.2 Les contours actifs sans bords

2.2.1 Le modèle de Mumford et Shah

Mumford et Shah [11] ont proposé un modèle de détection de contour basé sur une approche région.

Soit $\Omega \subset \mathbb{R}^2$ un ensemble ouvert et C une courbe paramétrique. Le problème de segmentation formulé par Mumford et Shah peut être vu de la façon suivante. On cherche à décomposer l'image $u_0 : \Omega \rightarrow \mathbb{R}$ en ensembles Ω_i tels que $\Omega = \cup_i \Omega_i \cup C$ (C étant les contours des Ω_i) et on cherche l'approximation optimale régulière par morceaux u de u_0 minimisant la fonction énergie

$$F^{MS}(u, C) = \mu \text{Length}(C) + \lambda \int |\nabla u|^2 dx dy + \int_{\Omega} |u_0(x, y) - u(x, y)|^2 dx dy \quad (1)$$

où $\mu > 0$ est un paramètre fixé pour pondérer le terme de régularisation. Ce terme de régularisation représente la longueur du contour C qui doit être minimale pour garantir un contour lisse. Plus μ sera grand, plus le contour sera lisse. L'autre terme est le terme d'attache aux données. Pour simplifier la fonction d'énergie, on choisit de restreindre l'approximation u à l'ensemble des fonctions constantes par morceaux, $u = c_i$ dans chaque ensemble Ω_i . Pour une courbe C fixée, l'énergie est minimisée pour $c_i = \text{moyenne}(u_0)$ dans Ω_i . On obtient la formule suivante :

$$E^{MS}(u, C) = \sum_i \int_{\Omega_i} |u_0 - c_i|^2 dx dy + \mu |C| + \lambda \int |\nabla u|^2 dx dy \quad (2)$$

Chan et Vese [2] ont ensuite mis en place la résolution de la minimisation de la fonction dans le cas où l'image u_0 est formée de deux régions d'intensités constantes et de valeurs distinctes c_1 pour l'objet et c_2 pour le fond. Pour calculer la minimisation, ils ont appliqués le formalisme level-set dont les bases ont été posées par Osher et Sethian (1988). Une courbe donnée C (figure 5) est représentée implicitement comme le niveau zéro d'une fonction de Lipschitz $\phi : \Omega \rightarrow \mathbb{R}$ telle que

$$\begin{aligned} \phi(x, y) &> 0 \text{ dans } \omega \\ \phi(x, y) &< 0 \text{ dans } \Omega \setminus \omega \\ \phi(x, y) &= 0 \text{ sur } C \end{aligned}$$

et qui, en pratique, est initialisée avec une fonction distance.

Dans le cas de l'étude de Chan et Vese (un objet et un fond), la fonction à minimiser est la suivante :

$$\begin{aligned} F(c_1, c_2, C) &= \mu \text{Length}(C) + \nu \text{Area}(\text{Inside}(C)) \\ &+ \lambda_1 \int_{\text{inside}(C)} |u_0(x, y) - c_1|^2 dx dy \\ &+ \lambda_2 \int_{\text{outside}(C)} |u_0(x, y) - c_2|^2 dx dy \end{aligned} \quad (3)$$

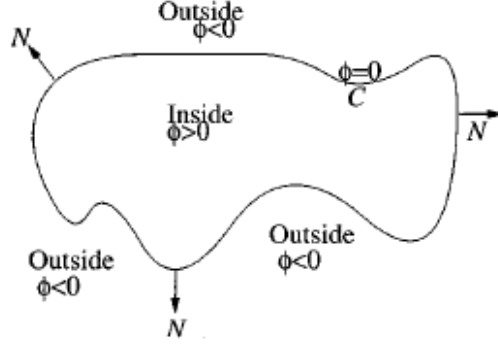


FIG. 5 – Courbe $C = \{(x, y), \phi(x, y) = 0\}$

Avec μ , ν , λ_1 et λ_2 des coefficients positifs respectivement pour les deux termes de régularisation et les deux termes d'attache aux données. Cependant, le deuxième terme de régularisation ne sera pas pris en compte dans la suite car celui-ci alourdit les calculs. On utilise ensuite la fonction de Heaviside dans la méthode des level-set définie par :

$$H(z) = \begin{cases} 1, & \text{if } z \geq 0 \\ 0, & \text{if } z < 0 \end{cases}$$

Et on définit la masse de Dirac par

$$\delta_0(z) = \frac{d}{dz}H(z)$$

Grâce à cette formulation, l'énergie devient

$$\begin{aligned} F(c_1, c_2, \phi) = & \mu \int_{\Omega} |\nabla H(\phi(x, y))| dx dy \\ & + \lambda_1 \int_{\Omega} |u_0(x, y) - c_1(\phi)|^2 H(\phi(x, y)) dx dy \\ & + \lambda_2 \int_{\Omega} |u_0(x, y) - c_2(\phi)|^2 (1 - H(\phi(x, y))) dx dy \end{aligned} \quad (4)$$

Les constantes c_1 et c_2 sont calculées pour un ϕ donné, respectivement comme la moyenne des pixels à l'intérieur du contour et à l'extérieur du contour.

Pour calculer le ϕ à l'itération suivante, on utilise l'équation d'Euler-Lagrange permettant ainsi de diriger le contour dans la direction de plus grande pente.

$$\frac{\partial \phi}{\partial t} = \delta_{\epsilon} \left[\mu \operatorname{div} \left(\frac{\nabla \phi}{|\nabla \phi|} \right) - \lambda_1 (u_0 - c_1)^2 + \lambda_2 (u_0 - c_2)^2 \right] \quad (5)$$

D'après l'équation 5, l'évolution de la fonction level-set ϕ (et donc du contour) est conditionnée par la comparaison de la valeur de l'image en un point donné, avec les valeurs moyennes existantes. Ainsi, ϕ est tirée vers les valeurs négatives

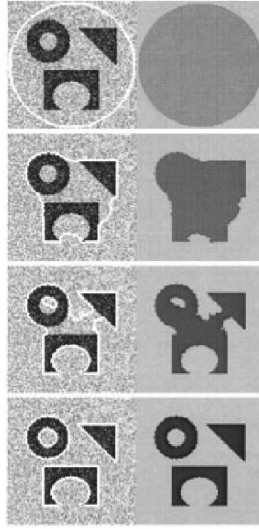


FIG. 6 – Segmentation d'un objet topologiquement complexe.[2]

si, au point considéré, l'image est plus proche de c_2 et vers les valeurs positives si elle est plus proche de c_1 . Cependant, si cette décision provoque une trop grande courbure de ϕ , le terme de régularisation peut jouer pour changer la décision.

Les tests effectués avec cette méthode donnent les résultats suivants

- On détecte des objets avec des contours à faible gradient.
- Il n'y a pas de problème lié à la topologie c'est-à-dire que le contour peut se scinder en plusieurs parties pour détecter plusieurs objets ayant la même intensité. (figure 6).
- On détecte des variations moyennes d'intensité à une échelle déterminée par le paramètre de régularisation.

2.2.2 La segmentation multiphase

A partir des travaux effectués précédemment sur la segmentation d'une image à deux phases, Chan et Vese ont généralisé le modèle pour une segmentation multiphase des images [14].

La construction des phases s'effectue de la manière suivante. On pose $\phi = (\phi_1, \dots, \phi_m)$ le vecteur fonction level set et $H(\phi) = (H(\phi_1), \dots, H(\phi_m))$ le vecteur fonction Heaviside. Chaque phase est déterminée par le signe de chaque level-set (figure 5). De par cette construction, on remarquera qu'avec n phases on aura besoin de seulement $m = \log(n)$ fonctions level set $\phi_i : \Omega \longrightarrow \mathbb{R}$. Prenons l'exemple

du cas de deux fonctions level-set. La fonction d'énergie correspondant à ce cas est

$$\begin{aligned}
F(c, \phi) = & \int_{\Omega} (u_0 - c_{11})^2 H(\phi_1) H(\phi_2) dx dy \\
& + \int_{\Omega} (u_0 - c_{10})^2 H(\phi_1) (1 - H(\phi_2)) dx dy \\
& + \int_{\Omega} (u_0 - c_{01})^2 (1 - H(\phi_1)) H(\phi_2) dx dy \\
& + \int_{\Omega} (u_0 - c_{00})^2 (1 - H(\phi_1)) (1 - H(\phi_2)) dx dy \\
& + \nu \int_{\Omega} |\nabla H(\phi_1)| + \nu \int_{\Omega} |\nabla H(\phi_2)|
\end{aligned} \tag{6}$$

Avec $c_{ij} = \text{moyenne}(u_0)$ dans la phase ij , $i, j = 0$ ou 1

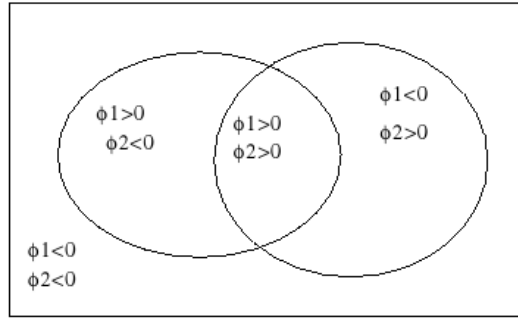


FIG. 7 – Les deux fonctions level-set partitionnent l'image en 4 régions : $\{\phi_1 > 0, \phi_2 > 0\}$, $\{\phi_1 > 0, \phi_2 < 0\}$, $\{\phi_1 < 0, \phi_2 > 0\}$, $\{\phi_1 < 0, \phi_2 < 0\}$.

En utilisant l'équation d'Euler-Lagrange on a

$$\begin{aligned}
\frac{\partial \phi_1}{\partial t} = & \delta_{\epsilon}(\phi_1) \left\{ \nu \operatorname{div} \left(\frac{\nabla(\phi_1)}{|\nabla(\phi_1)|} \right) - [((u_0 - c_{11})^2 - (u_0 - c_{01})^2) H(\phi_2) \right. \\
& \left. + ((u_0 - c_{10})^2 - (u_0 - c_{00})^2) (1 - H(\phi_2))] \right\}
\end{aligned} \tag{7}$$

$$\begin{aligned}
\frac{\partial \phi_2}{\partial t} = & \delta_{\epsilon}(\phi_2) \left\{ \nu \operatorname{div} \left(\frac{\nabla(\phi_2)}{|\nabla(\phi_2)|} \right) - [((u_0 - c_{11})^2 - (u_0 - c_{10})^2) H(\phi_1) \right. \\
& \left. + ((u_0 - c_{01})^2 - (u_0 - c_{00})^2) (1 - H(\phi_1))] \right\}
\end{aligned} \tag{8}$$

Nous avons l'équation dévolution des level-set, mais reste la question de leur initialisation. Chan et Vese ont conseillé une initialisation avec des cylindres répartis sur l'image comme vous pouvez le voir sur la première image de la figure 8.

Les résultats de cette méthode sont bons et permettent de segmenter correctement des images avec des objets d'intensités distinctes. Une application médicale

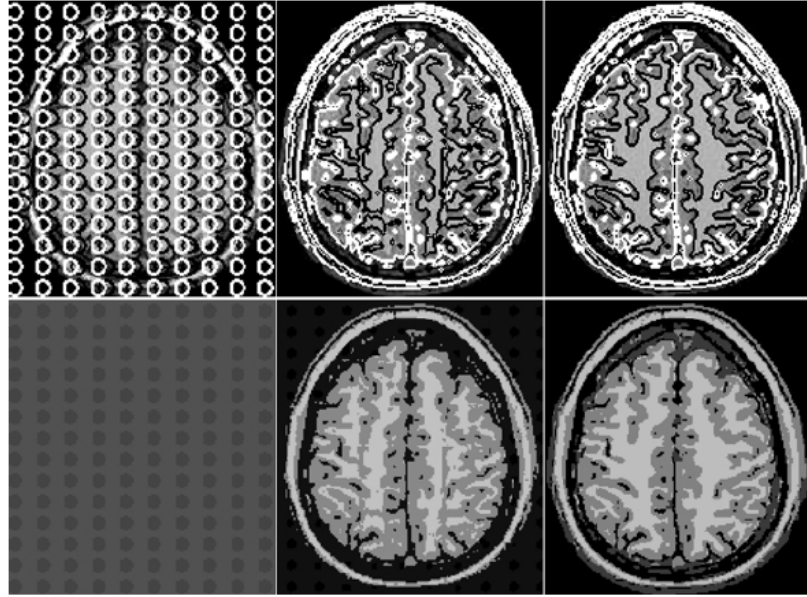


FIG. 8 – Evolution de la segmentation d’une image IRM de cerveau par la méthode de Chan et Vese en utilisant deux fonctions level-set (à gauche : initialisation, à droite : résultat final). Les images de la première ligne représentent les contours des fonctions level-set et les images de la deuxième ligne représente les régions obtenues. [14]

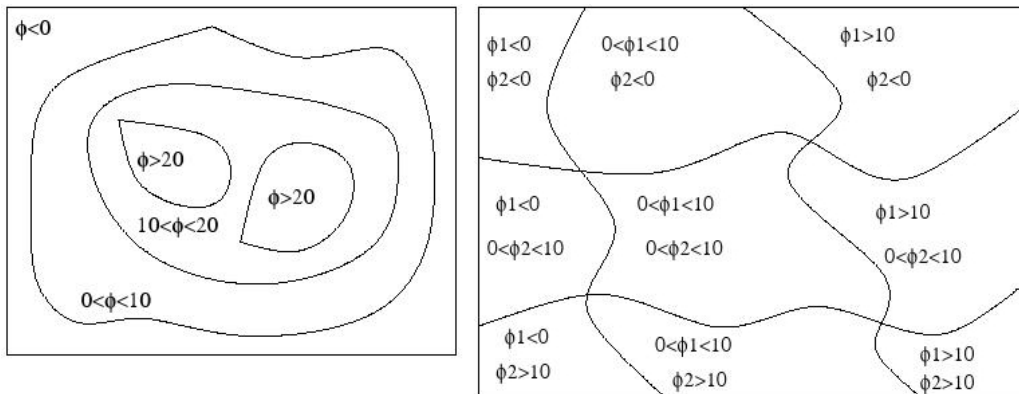


FIG. 9 – Gauche : les lignes de niveaux $x \in \Omega : \phi(x) = 0$, $x \in \Omega : \phi(x) = 10$ et $x \in \Omega : \phi(x) = 20$ divisent l’image en quatre phases disjointes. Droite : les lignes de niveaux $x \in \Omega : \phi_i(x) = 0$ et $x \in \Omega : \phi_i(x) = 10$ $i = 1, 2$ divisent l’image en 9 régions disjointes.[4]

consiste à segmenter les images IRM du cerveau (8) car celui-ci comporte plusieurs structures (essentiellement la matière grise, la matière blanche, les os et le LCR)

Une variante [4] de la méthode précédente a été étudiée notamment pour réduire la quantité de données à stocker et à traiter. Les phases sont déterminées à partir de plusieurs intervalles de valeurs pour les ϕ (et non $] -\infty, 0[$ et $[0, +\infty[$) (figure 7). Pour coder 9 phases, nous n'avons besoin que de 2 fonctions level-set alors qu'il en fallait 3 pour coder 8 phases. Nous ne détaillerons pas les équations notamment la fonction à minimiser et l'équation d'évolution des level-set car elles sont semblables aux équations précédentes. Les résultats obtenus par cette méthode sont équivalents à la méthode initiale tout en permettant de réduire la quantité des données.

Un autre type d'étude découlant de la segmentation multiphase de Chan et Vese a été conduit par Cremers [6] et [7]. Ces études portent sur un algorithme permettant de détecter un ou des objets partiellement cachés dont les contours sont connus et de détecter les contours des autres objets inconnus.

2.2.3 La segmentation multicanal : les images vectorisées

A partir du modèle de segmentation binaire (objet/fond) de Chan et Vese [3], des études ont été poursuivies afin de réaliser une segmentation multicanal [3] c'est-à-dire une segmentation à partir de plusieurs images recalées d'une même scène.

Soit $u_{0,i}$ le i -ème canal de l'image, $\overline{c^+} = (c_1^+, \dots, c_N^+)$ et $\overline{c^-} = (c_1^-, \dots, c_N^-)$ les deux vecteurs constants des valeurs moyennes $c_i^+ = \text{moyenne}(u_{0,i})$ pour $\phi \geq 0$ et $c_i^- = \text{moyenne}(u_{0,i})$ pour $\phi < 0$. Chan et Vese ont proposé une extension de leur modèle de segmentation au cas multicanal. La formulation en level set nous donne la fonction énergie à minimiser :

$$\begin{aligned} F(\overline{c^+}, \overline{c^-}, \phi) = & \mu \int_{\Omega} |\nabla \phi(x, y)| dx dy \\ & + \int_{\Omega} \frac{1}{N} \sum_{i=1}^N \lambda_i^+ |u_{0,i}(x, y) - c_i^+|^2 H(\phi(x, y)) dx dy \\ & + \int_{\Omega} \frac{1}{N} \sum_{i=1}^N \lambda_i^- |u_{0,i}(x, y) - c_i^-|^2 (1 - H(\phi(x, y))) dx dy \end{aligned} \quad (9)$$

A l'aide de l'équation d'Euler-Lagrange, à $\overline{c^+}$ et $\overline{c^-}$ fixé, on minimisera la fonction énergie précédente en faisant évoluer la fonction level-set ϕ en appliquant la formule suivante

$$\frac{\partial \phi}{\partial t} = \delta_{\epsilon} \left[\mu \operatorname{div} \left(\frac{\nabla \phi}{|\nabla \phi|} \right) - \frac{1}{N} \sum_{i=1}^N \lambda_i^+ (u_{0,i} - c_i^+)^2 + \frac{1}{N} \sum_{i=1}^N \lambda_i^- (u_{0,i} - c_i^-)^2 \right] \quad (10)$$

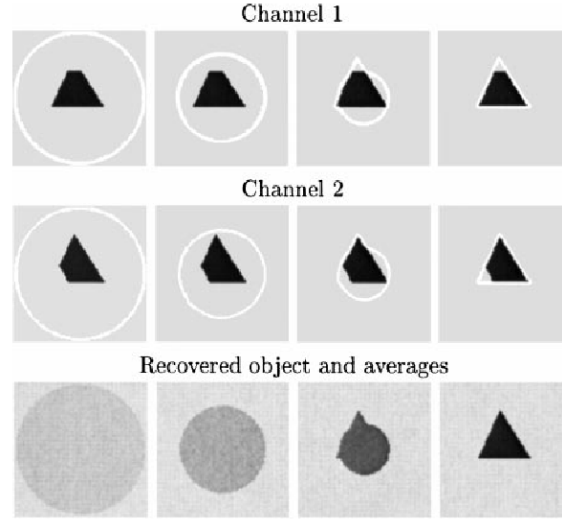


FIG. 10 – Il manque un bout du triangle sur chaque canal mais le contour final retrouve la totalité du triangle. [3]

Un résultat est donné sur la figure 10. On voit bien que grâce à la segmentation multicanal, on peut retrouver le triangle en totalité alors que des parties du triangle manquent sur chaque canal.

Cependant un problème de stabilité peut apparaître. En effet, la mesure d'hétérogénéité est une moyenne des mesures à travers les canaux. Avec cette définition, plusieurs courbes sont des minima de (13) et le modèle présente un risque d'oscillation entre ces solutions ou, en tout cas, une très grande dépendances aux conditions initiales.

A cause de ces problèmes, une nouvelle méthode dite logique a été établie sur la base de la précédente.

2.2.4 La segmentation multicanal : le modèle logique

Cette méthode [12] introduit une nouvelle analyse de l'image en prenant en compte le fait de chercher à savoir si un point appartient à l'objet de l'image. Pour cela, on définit les variables z_i^{in} et z_i^{out} :

$$z_i^{in}(u_0^i, x, y, C) = \begin{cases} 0 & \text{si } (x, y) \in C \text{ et } (x, y) \text{ est dans l'objet du canal } i \\ 1 & \text{sinon} \end{cases}$$

$$z_i^{out}(u_0^i, x, y, C) = \begin{cases} 1 & \text{si } (x, y) \notin C \text{ et } (x, y) \text{ est dans l'objet du canal } i \\ 0 & \text{sinon} \end{cases}$$

Ceci est une définition binaire puisque les z_i ne prennent pour valeur que 0 ou 1. On notera que 0 correspond à la valeur "vrai" car on cherche toujours à minimiser

une fonction. Dans la pratique, pour la mesure d'hétérogénéité, on utilisera les définitions suivantes permettant d'avoir une plage de valeurs comprises entre 0 et 1.

$$z_i^{in}(u_0^i, x, y, C) = \frac{|u_0^i(x, y) - c_+^i|^2}{\max_{(x,y) \in u_0^i} u_0^i{}^2}$$

$$z_i^{out}(u_0^i, x, y, C) = \frac{|u_0^i(x, y) - c_-^i|^2}{\max_{(x,y) \in u_0^i} u_0^i{}^2}$$

Grâce à ces opérations logiques, on peut définir l'intersection et l'union de deux images en traitant différemment les points qui sont dans l'objet et ceux qui sont en dehors de l'objet. Prenons l'exemple de l'union. Si un point (x,y) est dans l'objet dans au moins un canal alors ce point est dans l'objet final. Pour l'intersection, un point est considéré dans l'objet si et seulement si ce point est à l'intérieur de l'objet dans tous les canaux.

Pour interpoler l'union et l'intersection des variables z_i , les formules suivantes ont été proposées :

$$f_{\cup} = \sqrt{z_1 \cdot z_2} \quad \text{pour l'union}$$

$$f_{\cap} = 1 - \sqrt{(1 - z_1)(1 - z_2)} \quad \text{pour l'intersection}$$

L'union des objets est la combinaison de l'union des intérieurs des objets et l'intersection des points en dehors des objets. De même l'intersection des objets est obtenue en combinant l'intersection des intérieurs des objets et l'union des extérieurs des objets. On a donc les deux formules suivantes

$$f_{A1 \cup A2}(x, y) = \sqrt{z_1^{in}(x, y)z_2^{in}(x, y)} + 1 - \sqrt{(1 - z_1^{out}(x, y))(1 - z_2^{out}(x, y))}$$

$$f_{A1 \cap A2}(x, y) = 1 - \sqrt{(1 - z_1^{in}(x, y))(1 - z_2^{in}(x, y))} + \sqrt{z_1^{out}(x, y)z_2^{out}(x, y)}$$

Si on a n canaux, il suffit de combiner les opérations logiques entre les différents canaux. On peut notamment donner comme exemple l'union des objets

$$f_{A1 \cup \dots \cup A_n} = \left(\prod_{i=1}^n z_i^{in} \right)^{1/n} + 1 - \left(\prod_{i=1}^n (1 - z_i^{out}) \right)^{1/n}$$

Nous pouvons maintenant écrire la fonction énergie à minimiser pour une combinaison $f(z_1^{in}, z_1^{out}, \dots)$ quelconque des canaux

$$F(\phi, c^+, c^-) = \mu |C(\phi)| + \lambda \int_{\Omega} f_{in}(z_1^{in}, \dots, z_n^{in}) H(\phi) \\ + f_{out}(z_1^{out}, \dots, z_n^{out}) (1 - H(\phi)) dx dy$$

Les résultats expérimentaux permettent de mettre en valeur plusieurs atouts pour cette méthode

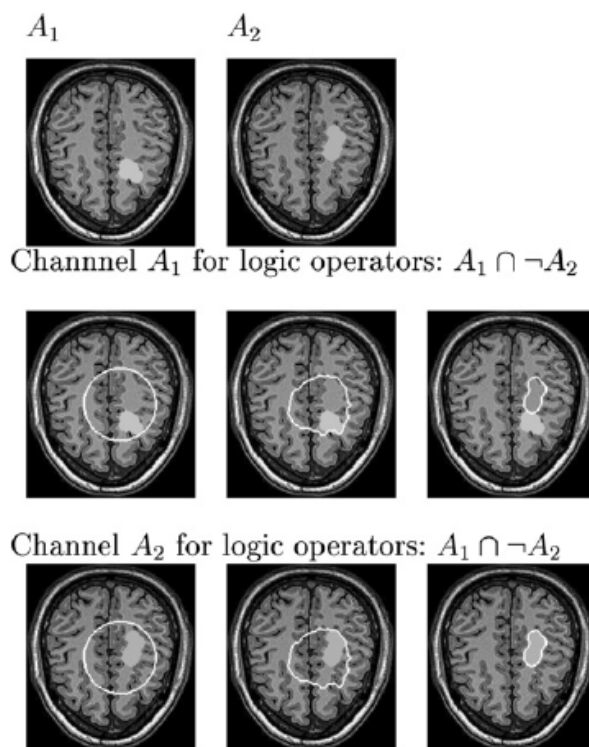


FIG. 11 – Détection d'anomalies fictives dans le cerveau. On peut trouver les anomalies du canal 1 qui ne sont pas dans le canal 2 et inversement.[12]

- La segmentation reste basée sur une approche région ce qui implique une détection des objets à faible gradient, topologiquement complexe ou encore d'objets 3D.
- L'utilisateur peut choisir quelles opérations (intersection, union) l'algorithme doit effectuer sur chaque canal (figure 8).
- En imagerie médicale, cette méthode permet d'utiliser les images provenant de divers imageurs comme par exemple les images T1 et T2 d'un examen IRM.

3 Algorithmes de segmentation multiphase et multicanal par minimisation de l'énergie de Mumford-Shah

3.1 Méthode de Chan et Vese

La généralisation de la méthode de Chan et Vese au cas multiphase a été codée par Elsa Angelini et son extension au multicanal par Vincent Israel-Jost [8]. Je précise que mon travail sur cet algorithme a été de le comprendre et de faire des tests pour trouver les paramètres permettant de segmenter au mieux une image IRM du cerveau.

3.1.1 Segmentation multiphase et permutations

Avec la formulation en level-set, l'algorithme de segmentation multiphase choisit, pour chaque point, la meilleure phase selon l'équation d'évolution de ϕ (équation 7 et 8 page 18). Prenons un exemple pour comprendre comment évolue cette fonction ϕ . Supposons qu'en un point de l'image, on ait $H(\phi_1) = 0$ ($\phi_1 < 0$) et $H(\phi_2) = 1$ ($\phi_2 > 0$), le point appartient donc à la phase (0,1). Selon l'équation d'évolution de ϕ_1 (équation 7), l'algorithme choisit la meilleure phase entre la phase (1,1) ou la phase courante (0,1). De même, selon l'équation d'évolution de ϕ_2 (équation 8), l'algorithme choisit la meilleure phase entre la phase (0,0) ou la phase courante (0,1). On peut dessiner les diagrammes montrant les évolutions possibles des phases (figure 12(a)). On remarque que chaque phase n'est en compétition qu'avec deux autres et donc qu'elle ne "voit" pas la troisième. Ainsi pour passer de la phase (0,0) à la phase (1,1), il faut passer par la phase (1,0) ou (0,1). Or, passer par une de ces phases intermédiaires ne peut se faire que si on minimise l'énergie en passant par ces phases, ce qui est rarement le cas. Ainsi, si un point doit appartenir à (0,0) et est initialisé à (1,1), alors il reste généralement dans cette phase (1,1) et ne va pas évoluer vers la bonne phase (0,0).

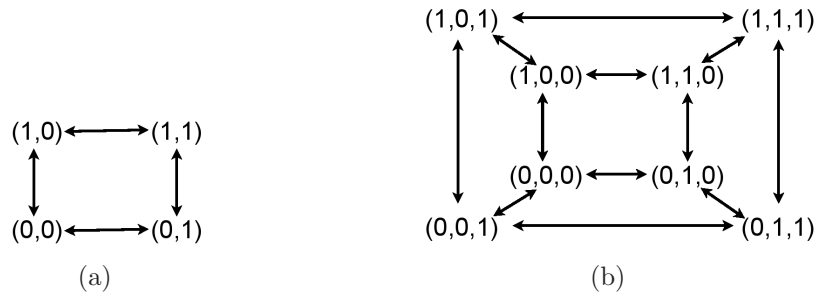


FIG. 12 – Diagramme d'évolution des phases à 2 et 3 level-set

Le raisonnement est le même pour une segmentation à trois fonctions level-set (figure 12(b)). On se rend bien compte que plus le nombre de level-set augmente, plus le nombre de phases "invisibles" augmente et cela réduit d'autant l'efficacité

de l'algorithme puisqu'il faut augmenter le nombre de permutations.

Une solution, celle qui a été retenue [8], est de permuter les phases afin que chaque phase soit en compétition avec toutes les autres phases pendant un certain temps de l'algorithme. Prenons le cas de deux fonctions level-set. Si, au cours de l'algorithme, on permute les phases (1,0) et (0,0) alors la phase (1,0) sera en compétition avec (0,1) et la phase (0,0) sera en compétition avec la phase (1,1). Pour faire cela, il suffit de changer le signe de ϕ_1 dans les zones où $\phi_2 = 0$. Les permutations sont faites après un nombre fixe S d'itérations.

Ces permutations sont aussi implémentées pour le cas 8 phases. On peut voir sur le diagramme (figure 12(b)) deux anneaux de phases, l'anneau intérieur avec les phases en $(\dots, 0)$ et l'anneau extérieur en $(\dots, 1)$. On réalise d'abord une rotation complète de l'anneau intérieur afin que toutes les phases en $(\dots, 0)$ aient été en compétition avec les phases en $(\dots, 1)$. on réalise ainsi $3S$ itérations. Puis on effectue deux permutations du même type que dans le cas à quatre phases afin que, sur un anneau, les phases qui ne se sont pas encore vues, comme les phases (1,1,0) et (0,0,0) ou (1,1,1) et (0,0,1), soient en compétition. Cela rajoute $2S$ itérations. Un cycle de permutation nécessite donc $5S$ itérations. Le nombre de cycle est ensuite choisi par l'utilisateur.

3.1.2 Fusion des données et segmentation multicanal

Dans le cas d'une segmentation multiphase et multicanal, nous ne sommes pas dans le cas d'une segmentation objet/fond. En effet, lorsque l'image est binaire, il est aisé de déterminer le fond et l'objet et ainsi de pouvoir fusionner les images en utilisant les formules d'union ou d'intersection (page 20). En multiphase, il existe plusieurs objets. On généralise la formule de la mesure d'hétérogénéité z_i^j par :

$$z_i^j(M) = \frac{|u_0^i(M) - c_i^j|^2}{\text{contraste}(u_0^i)^2} \quad (11)$$

avec M un point de Ω , c_i^j la moyenne dans le canal i de l'image u_0^i à l'intérieur de la phase j et $\text{contraste}(u_0^i(P))$ le contraste de l'image donné par la formule suivante ci-dessous. La mesure d'hétérogénéité est toujours comprise dans l'intervalle $[0 \ 1]$.

$$\text{contraste}(u_0^i) = \max(u_0^i) - \min(u_0^i) \quad (12)$$

La règle de fusion que nous avons choisie est une règle d'intersection des phases et nous allons détailler les raisons qui ont motivé notre choix.

Un élément fondamental à prendre en compte est le fait que les résultats que nous devons obtenir ne doivent pas être contingents c'est à dire que nous devons isoler des objets facilement interprétable. En regardant la figure 3.1.2, représentant la segmentation d'une image multicanal composée de trois régions distinctes dans chaque canal avec un fond commun et un rectangle central décomposé en deux sous-objets, nous montrons que la règle d'union des objets n'a pas le comportement escompté.

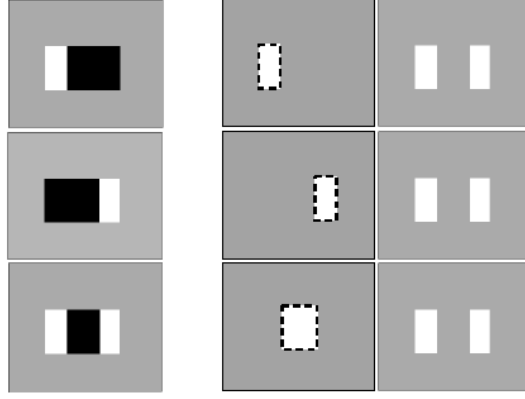


FIG. 13 – Première colonne (du haut vers le bas) : images du canal 1 et canal 2 formant trois 4 régions distinctes (dernière image) dont les correspondances des couleurs sont, blanc=100, noir=0, gris=10. Deuxième colonne : choix de trois initialisations différentes aboutissant au même résultat de segmentation (troisième colonne)

On a donc 4 sous régions qui ont pour couple de valeurs dans le canal 1 et 2 : (10,10), (0,100), (0,0) et (100,0). Pour une segmentation à deux phases, nous avons essayé plusieurs initialisations prenant différentes parties du rectangle central. Nous avons suivi les recommandations de Sandberg [12] en appliquant l'union sur la phase objet et l'intersection sur la phase utilisée pour le fond. Le résultat est le même quelle que soit l'initialisation, la région (0,0) a été classée dans la phase représentant le fond bien qu'elle ait été classée dans la phase objet au cours des itérations. Ceci s'explique par le fait que l'union permet de combiner des régions qui ont des valeurs moyennes très hétérogènes. Il est surprenant de voir que dans la phase objet initialisée à (100,0), on retrouve la région (0,100) pourtant plus éloignée de la région (0,0). Le manque de stabilité et le manque de prédictibilité d'évolution font que nous ne choisissons pas l'union comme règle de fusion.

Nous allons choisir une règle d'intersection même si celle-ci ne détermine pas explicitement les cas conflictuels. Avec une règle d'intersection sur toutes les phases, il est nécessaire d'avoir 4 phases pour segmenter notre image test puisque nous segmentons des régions qui sont homogènes. La règle d'intersection que nous choisissons est le maximum et nous avons donc l'énergie suivante à minimiser :

$$E = \sum_{j=1}^P \int_{R_j} \max_{i=1,\dots,n} (z_i^j(x)) dx + \mu |C| \quad (13)$$

avec n le nombre de canal, $|C|$ la longueur des contours des phases R_j , P le nombre de phases et z_i^j la mesure d'hétérogénéité de la phase j dans la canal i dont la formule à l'équation 11.

3.1.3 Résultats

Pour comprendre le fonctionnement de l'algorithme, nous allons étudier l'influence de chaque paramètre afin de dégager les paramètres optimaux. Ces pa-

ramètres sont optimaux pour l'image qui a été traitée. Comme nous allons traiter des images du même type acquises dans des conditions similaires, nous pouvons penser que les paramètres peuvent être appliqués à toute image d'IRM cérébrale.

Influence du paramètre de régularisation Le paramètre de régularisation permet de lisser les contours des phases. Il est important d'avoir un paramètre qui permet de conserver une forme précise des structures tout en éliminant les petites irrégularités. Lorsque ce coefficient est petit, les phases sont bruitées alors que lorsqu'il est grand, les phases ont des bords lisses et ne sont pas bruitées. Visuellement (figure 14), on peut dire que la meilleure segmentation est obtenue pour un μ égal à 0.1. Avec ce coefficient, on obtient des phases peu bruitées et on garde la forme de toutes les structures notamment les noyaux qui sont tronqués avec d'autres valeurs de régularisation.

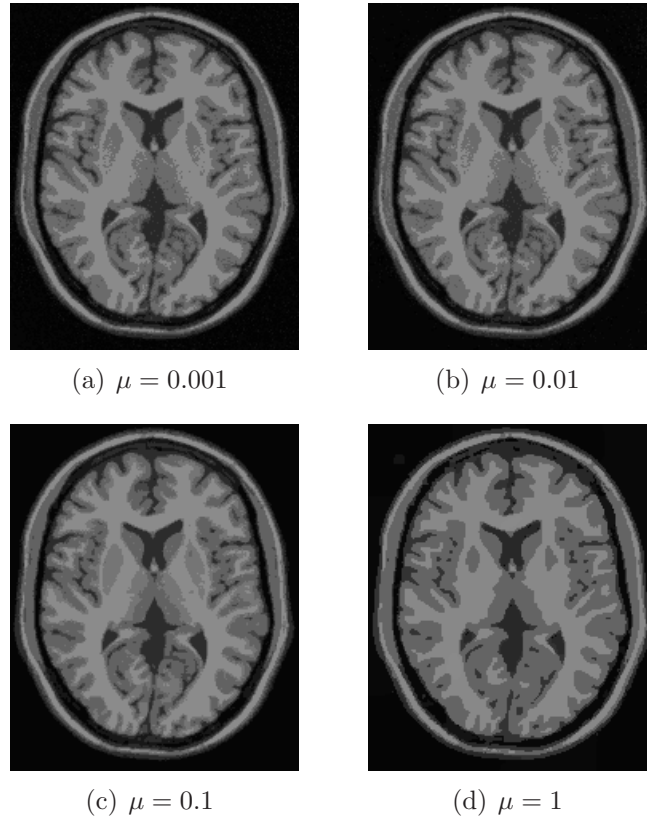


FIG. 14 – Segmentation de la coupe 80 de Brainweb avec différents coefficients de régularisation.

Influence du nombre d'itérations inter-permutation L'algorithme nécessitant beaucoup de temps de calcul, il est nécessaire de faire des tests pour savoir si on peut réduire le nombre d'itérations d'autant qu'on ne peut pas connaître mathématiquement le nombre optimal d'itérations pour minimiser au mieux la fonction énergie. Voici les résultats de la segmentation avec plusieurs nombres d'itérations inter-permutation (noté st) (Figure 15) et un μ de 0.1 qui est la valeur

optimale obtenue dans le paragraphe précédent.

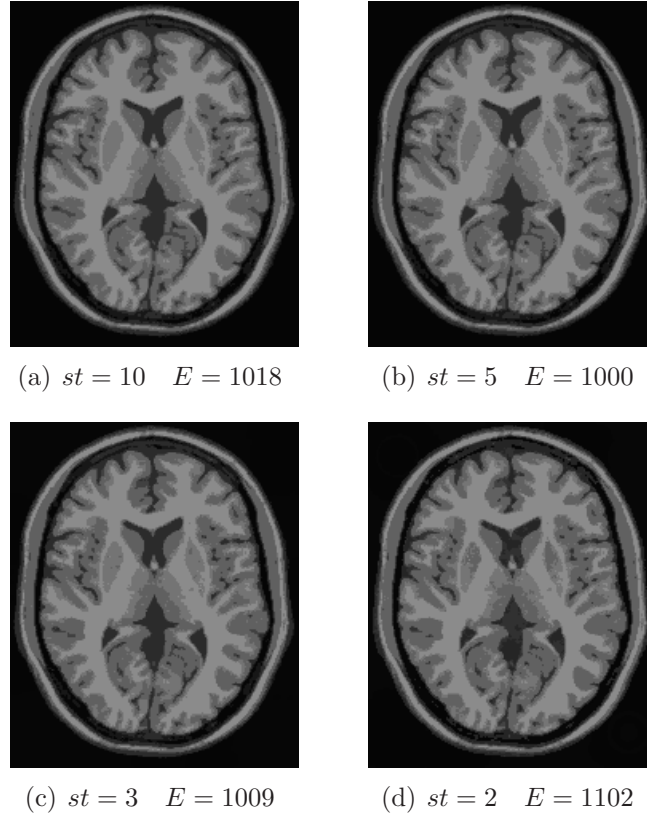


FIG. 15 – Segmentation de la coupe 80 de Brainweb avec un nombre différent d'itérations entre les permutations

Ces résultats amènent plusieurs remarques. Tout d'abord, nous pouvons dire que lorsque $st = 2$, la segmentation est mauvaise. En effet, deux phases représentent le LCR et deux phases pour les noyaux. Cette mauvaise segmentation est confirmée par une valeur finale de l'énergie bien plus élevée que pour les trois autres cas. Visuellement, on remarque que les résultats sont proches pour les 3 autres cas ($st = 3, 5$ et 10). Ceci est confirmé par le calcul de l'énergie qui montre bien que si on laisse 3, 5 ou 10 itérations entre les permutations, la valeur finale de l'énergie est très peu modifiée. On n'a qu'une variation de 2% entre les valeurs finales des énergies pour ces trois cas. On choisira donc 3 itérations entre les permutations.

Influence de la réinitialisation Nous avons vu qu'après chaque permutation, nous réinitialisons les fonctions level-set. Cela empêche les fonctions level-set d'atteindre des valeurs plus élevées. Supposons que la fonction level-set soit constante de valeur 500 sur un intervalle quelconque (Figure 16). Si dans cet intervalle, une partie de la fonction doit passer négative à cause d'une permutation, on va donc avoir sur ce même intervalle, une transition brusque de 500 à -500. Si un point situé à 500 doit passer en négatif à cause de l'évolution de ϕ , la distance pour atteindre les valeurs négatives est longue et on a donc besoin de plus d'itérations pour que le point se déplace dans la bonne phase. Pour résoudre ce problème, on

choisit de réinitialiser les fonctions level-set après chaque permutation afin d'avoir des variations plus petites. Nous pouvons alors nous demander si réinitialiser à chaque itération améliore la segmentation.

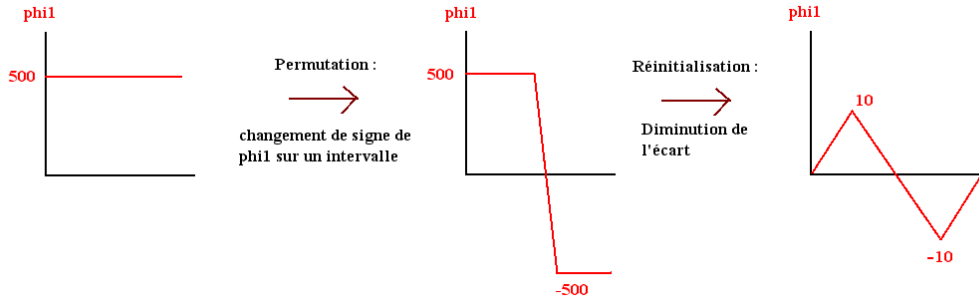


FIG. 16 – Effet de la réinitialisation sur les level-set.

Comme nous pouvons le voir sur la figure 17, la réinitialisation à chaque itération a pour effet de lisser les phases et aussi d'utiliser moins de phases pour segmenter l'image. Au vu de la valeur de l'énergie, on peut se dire qu'il est préférable de réinitialiser les level-set à chaque itération, mais visuellement l'image de gauche est mieux segmentée notamment au niveau des noyaux. La diminution de l'énergie est due en réalité au fait que l'algorithme va utiliser moins de phases ce qui réduit l'énergie de longueur.

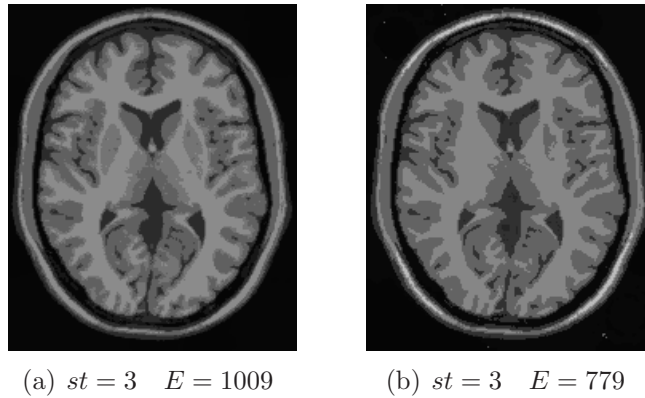


FIG. 17 – Segmentation de la coupe 80 de Brainweb avec réinitialisation après chaque permutation (à gauche) et à chaque itération (à droite)

Influence du nombre de cycles de permutation Nous avons vu qu'un cycle de permutation nécessitait $5S$ itérations. Dans notre cas, avec S valant 3, il faut donc 15 itérations par cycle. Les résultats précédents ont été réalisés avec 2 cycles de permutations. Nous avons lancé l'algorithme avec 1,2,3 et 4 cycles de permutation (Figure 18).

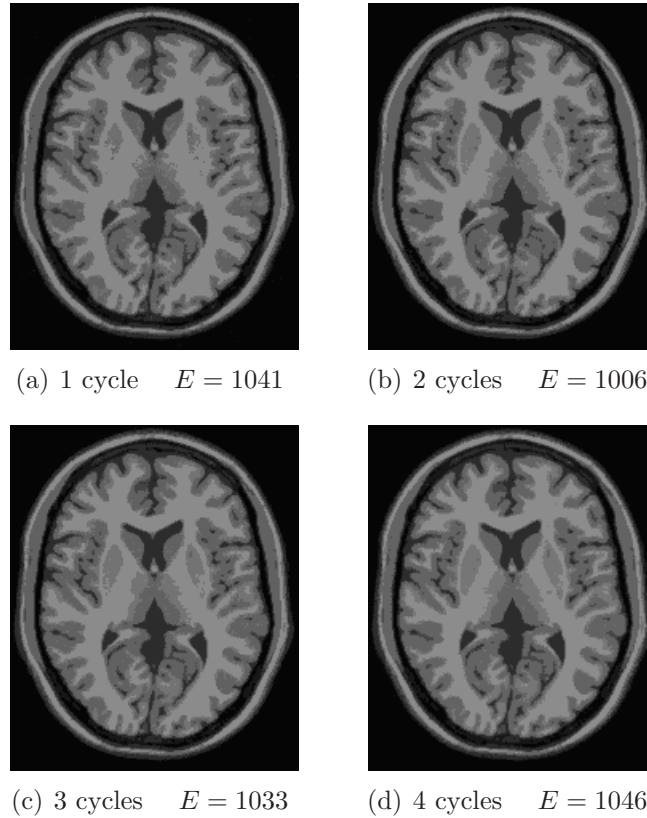


FIG. 18 – Segmentation de la coupe 80 de Brainweb avec un nombre différent de cycles de permutations. ($\mu = 0.1$, $S = 3$ et on a 20 itérations après la dernière permutation)

Nous pouvons voir qu'avec deux cycles, l'énergie est la plus petite et la segmentation est bonne. Avec 4 cycles, le résultat est visuellement meilleur mais le temps de calcul a été doublé par rapport à deux cycles ce qui est un inconvénient. Au vu des résultats et en prenant en considération l'énergie, le temps de calcul et l'appréciation visuelle, nous pouvons dire que le nombre de cycles optimal est 2.

Conclusion des tests Rappelons que les valeurs obtenues précédemment sont optimales pour l'image qui a été utilisée mais nous pouvons penser qu'ils peuvent être une base pour segmenter d'autres images du même type. Grâce à tous ces tests, on peut dire que la segmentation est optimale pour les paramètres suivants

- $\mu = 0.1$
- $S = 3$
- Réinitialisation après chaque permutation
- 2 cycles
- 30 itérations

Une courte étude équivalente à la précédente pour l'algorithme de Chan et Vese à 4 phases a également été réalisée. Il en est ressorti les valeurs suivantes.

- $\mu = 0.15$

- $S = 10$
- Réinitialisation après chaque permutation
- 3 cycles
- 40 itérations

Limites de la méthode L'algorithme de minimisation de la fonction énergie par Chan et Vese donne des résultats visuellement bons. Cependant, nous pouvons voir les limites de l'algorithme. Premièrement, l'algorithme en lui-même est complexe et donc long à mettre en oeuvre. Il nécessite des calculs de carte de distance pour réinitialiser les fonctions level-set, des opérations sur les phases pour les permutations, des calculs complexes pour évaluer la régularisation. Deuxièmement, à cause de ces calculs complexes, les temps de calculs sont longs. Enfin, la notation utilisée pour représenter les phases par les levels-set ne permet de représenter qu'un nombre fixe de phases. En effet, avec 2 level-set on a 4 phases, avec 3 level-set, on a 8 phases. Il n'est pas toujours nécessaire d'avoir 8 phases pour bien segmenter une image, on peut n'avoir besoin que de 6 phases ou bien de 9 phases. Il arrive souvent que le fond soit segmenté par deux phases de même valeur alors qu'il n'en faudrait qu'une. Bien que les images médicales puissent être segmentées avec un nombre de phases inférieur à 8, l'implémentation de cet algorithme pour 16 phases augmente considérablement la complexité de l'algorithme, en particulier la partie concernant les permutations des phases.

A cause de ces problèmes, nous allons utiliser des méthodes d'optimisation plus simples et plus rapides et vérifier leurs compatibilités pour traiter des images médicales. Nous allons notamment chercher des méthodes faisant intervenir les champs de Markov.

3.2 Méthode du recuit simulé

3.2.1 Principe théorique

Modélisation probabiliste de l'image L'image est formée d'un ensemble fini S de site s_i correspondant ici aux pixels. On associe à chaque site un descripteur qui sera dans notre cas une étiquette prenant des valeurs dans $E=[1 \text{ NP}]$. Pour introduire une interaction locale, on définit un système de voisinage Υ .

$$\Upsilon_s = \{t\} \quad \text{tels que} \quad \begin{cases} s & \in \Upsilon_s \\ t & \in \Upsilon_s \Rightarrow s \in \Upsilon_t \end{cases} \quad (14)$$

A partir de ce système de voisinage, on définit un système de cliques C qui est soit un singleton, soit un ensemble de pixels voisins. Dans la suite, la clique notée c sera, en dimension 2, la 4 connexité ou la 8 connexité. On quantifie cette interaction à l'aide du potentiel U_c et nous pouvons donc définir l'énergie globale de l'image comme la somme des potentiels de toutes les cliques

$$U = \sum_{c \in C} U_c \quad (15)$$

L'image dont on dispose est considérée comme une réalisation d'un champ aléatoire. On associe à un site s , une variable aléatoire X_s prenant ses valeurs dans E . Le niveau de gris x_s est donc une réalisation de la v.a. X_s . L'image est considérée comme une réalisation x du champ X .

Champs de Markov Soit x_s la valeur du descripteur prise au site s et $x_s = (x_t)_{t=s}$. la configuration exceptée le site s . On définit le champ de Markov de la manière suivante :

X est un champ de Markov si et seulement si la probabilité conditionnelle locale en un site n est fonction que de la configuration du voisinage du site considéré. Ainsi le niveau de gris en un site ne dépend que des niveaux de gris des pixels voisins de ce site. Il nous faut maintenant avoir accès à ces probabilités conditionnelles ce qui est possible grâce aux champs de Gibbs.

Champs de Gibbs Une mesure de Gibbs de fonction d'énergie $U : \Omega \rightarrow \mathbb{R}$ est la probabilité P définie par :

$$P(X = x) = \frac{1}{Z} \exp(-U(x)) \quad (16)$$

$$U(x) = \sum_{c \in C} U_c(x) \quad (17)$$

L'algorithme de Métropolis Cet algorithme a été mis au point dans les années 50 et permet de construire une suite d'images qui suivent la loi du champ de Markov après un nombre suffisamment grand d'itérations. Voici les opérations effectuées à l'étape n :

- choix d'un site s

- On tire aléatoirement un descripteur λ dans E selon une loi uniforme.
- Calcul de la variation d'énergie ΔU pour le passage du site s de $x_s^{(n-1)}$ à λ .
 - si $\Delta U < 0$, le changement est accepté : $x_s^{(n)} = \lambda$
 - si $\Delta U > 0$, le changement est accepté selon une probabilité $p = \exp(-\Delta U)$

Le recuit simulé Grâce à l'algorithme de Métropolis, nous pouvons échantillonner selon la loi de probabilité de Gibbs associé au champ de Markov. Nous voulons calculer la ou les configurations les plus probables qui correspondent aux états d'énergie minimale, car n'oublions pas que le but final est toujours de minimiser la fonction énergie.

Cet algorithme est basé sur un paramètre de température T . Une distribution de Gibbs avec un paramètre de température est une probabilité :

$$P_T(X = x) = \frac{1}{Z(T)} \exp\left(-\frac{U(x)}{T}\right) \quad (18)$$

avec $Z(T) = \sum_x \exp(-\frac{U(x)}{T})$ On remarque que lorsque T tend vers l'infini, P_T converge vers une probabilité uniforme sur Ω ce qui signifie que pour une température infinie, tous les états sont équiprobables. De plus lorsque T tend vers 0, P_T est uniformément distribué sur les minima globaux de l'énergie c'est à dire sur les configurations les plus probables. C'est ce résultat qui est à la base de l'algorithme du recuit simulé. Ce dernier permet de rechercher une configuration d'énergie minimale d'un champ de Gibbs.

L'algorithme est le suivant

- choix d'une température initiale $T^{(0)}$ suffisamment élevée.
- choix d'une configuration initiale quelconque $x^{(0)}$
- à l'étape n
 - On choisit une configuration $x^{(n)}$ pour la loi de Gibbs $\frac{U(x)}{T^{(n)}}$ à partir de la configuration $x^{(n-1)}$ en utilisant l'algorithme de Métropolis.
 - On diminue lentement la température $T^{(n)} > \frac{c}{\log(1+n)}$.
 - on passe à l'itération suivante

Cette méthode nous garantit de converger vers un minimum global car il accepte les remontées d'énergie. Avec la décroissance de la température, ces sauts énergétiques sont supprimés au fur et à mesure. La descente en température doit donc se faire lentement pour que l'algorithme ne soit pas piégé dans un minimum local.

3.2.2 Pratique

Nous avons vu dans le paragraphe précédent la théorie du recuit simulé. Voyons maintenant comment coder en pratique cet algorithme. L'énergie que nous cherchons à minimiser est celle de Mumford-Shah (Equation 2) qui contient deux termes, l'attache aux données et la régularisation.

L'algorithme est le suivant

- On génère une image d'étiquettes aléatoire
- On parcourt l'image pixel par pixel :
- Calcul de l'énergie E_n (attache aux données + régularisation) si on remplace le pixel par la nouvelle étiquette
- Calcul de la différence d'énergie au pixel entre l'énergie nouvelle et l'ancienne énergie $\Delta E = E_n - E_{n-1}$
 - On accepte la transition si au pixel considéré $\Delta E \leq 0$
 - sinon : génération aléatoire de ρ selon une loi uniforme sur $[0,1]$
 - Si $\rho \leq \exp(-\frac{\Delta E}{T})$ on accepte la transition
 - Sinon on garde l'ancienne étiquette.
- On diminue la température $T_{n+1} = \alpha T_n$ avec α très proche de 1 (0,99 par exemple).
- calcul des nouvelles valeurs moyennes des phases.

L'algorithme se termine lorsque le nombre de changements par itération est petit. Les valeurs moyennes des phases sont initialisées soit automatiquement soit manuellement.

Cet algorithme autorise les remontées d'énergie pour ne pas être piégé dans un minimum local et permet donc de converger vers le minimum global. Nous savons qu'avec une telle méthode les temps de calcul seront très long. Cet algorithme nous permettra uniquement de se rapprocher le plus possible du minimum global et de pouvoir comparer les résultats des algorithmes par rapport à ce minimum global.

3.2.3 La régularisation

La régularisation utilise le modèle de Potts qui est particulièrement recommandé pour les images d'étiquettes. Nous allons voir sur un exemple comment celui-ci fonctionne dans le cas du recuit simulé. Le but est de réduire au maximum la longueur des phases et donc plus un pixel sera entouré d'une même phase plus celui-ci aura de chance d'appartenir à cette même phase. Pour cela, on calcule pour chaque pixel, le nombre de cotés qui correspondent à des frontières, c'est à dire le nombre de pixels qui sont différents du pixel considéré. Supposons qu'à la fin de l'itération $n - 1$, nous ayons obtenu sur une partie de l'image ces valeurs

$$Image_{(n-1)} = \begin{pmatrix} 4 & 1 & 1 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{pmatrix}$$

Au début de l'itération n , on tire une image aléatoirement et on tire entre autre 1 pour le pixel central. Si on s'intéresse au pixel central, on construit la matrice $Intermediaire_{(n)}$ qui est la matrice $Image_{(n-1)}$ avec la phase tirée aléatoirement qui est 1. On obtient donc la matrice suivante où seul le pixel central est modifié.

$$Intermediaire_{(n)} = \begin{pmatrix} 4 & 1 & 1 \\ 2 & 1 & 2 \\ 2 & 2 & 2 \end{pmatrix}$$

A partir de celle-ci, on forme une matrice $A_{(n)}$ telle qu'on ait 1 sur les pixels qui ont une phase différente de la phase centrale et 0 sinon, ce qui donne

$$A_{(n)} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

Il faut maintenant calculer la régularisation avec ce changement de phase. Le principe est le suivant. On parcourt les pixels voisins et on incrémente d'une valeur donnée l'énergie (1 pour les pixels de la 4-connexité et $\frac{1}{\sqrt{2}}$ pour les pixels de la 8-connexité) si ce pixel est différent de la phase du pixel central. On remarquera que ce calcul s'apparente à une convolution. En effet, il suffit de convoluer l'image A_n par la matrice R suivante

$$R = \begin{pmatrix} \frac{1}{\sqrt{2}} & 1 & \frac{1}{\sqrt{2}} \\ 1 & 0 & 1 \\ \frac{1}{\sqrt{2}} & 1 & \frac{1}{\sqrt{2}} \end{pmatrix}$$

Les coefficients sur les pixels diagonaux sont volontairement inférieurs à 1 car ils sont situés à une distance plus grande et doivent donc avoir un poids plus faible. On choisit $\frac{1}{\sqrt{2}}$ car on considère que la distance entre deux pixels diagonaux est $\sqrt{2}$ qui est la demi diagonale d'un carré. Le résultat de la convolution donne $3 + 3\sqrt{2}$. Cette valeur est ensuite normalisée par $4 + 4\sqrt{2}$ afin d'avoir une énergie de régularisation par pixel qui vaut 1 si le pixel est uniquement entouré par des phases différentes de la sienne et 0 s'ils appartiennent tous à sa phase. Ceci est illustré sur la figure 19. Pour les images (a) et (b), le noir signifie que le pixel appartient à la phase 1 et le blanc à la phase 2. Pour l'image (c), le noir signifie que l'énergie de régularisation est nulle et le blanc que l'énergie de régularisation est maximale et vaut 1.

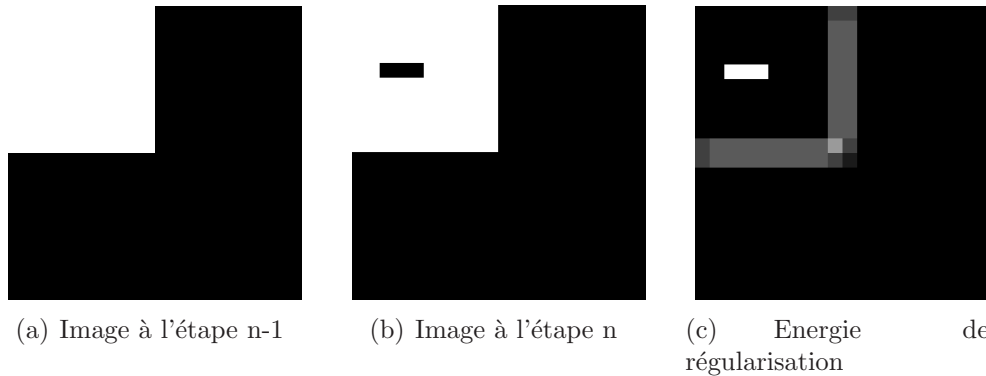


FIG. 19 – Calcul de l'énergie de régularisation (c). Plus un pixel est entouré de phases différentes de la sienne, plus l'énergie de régularisation sera élevée (blanc). A l'inverse si un point est uniquement entouré de pixels de sa phase alors l'énergie est nulle (noir).

Dans le cas d'un traitement tridimensionnel, la régularisation s'effectue en appliquant le filtre ci-dessous de taille $3 \times 3 \times 3$ et de normaliser par un coefficient qui est la somme des termes de cette matrice $\frac{8}{\sqrt{3}} + \frac{12}{\sqrt{2}} + 6$.

$$\begin{pmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{2}} & 1 & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} \end{pmatrix} \quad \begin{pmatrix} \frac{1}{\sqrt{2}} & 1 & \frac{1}{\sqrt{2}} \\ 1 & 0 & 1 \\ \frac{1}{\sqrt{2}} & 1 & \frac{1}{\sqrt{2}} \end{pmatrix} \quad \begin{pmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{2}} & 1 & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} \end{pmatrix}$$

3.2.4 Résultats

Les différents paramètres sur lesquelles nous pouvons jouer sont le paramètre de régularisation μ , le nombre de phases, la température initiale, le coefficient α . Parmi tous ces paramètres, on n'étudiera que l'influence du paramètre de régularisation. En effet, les autres paramètres ne vont pas changer pour plusieurs raisons. Tout d'abord, afin de comparer les méthodes entre elles, nous choisirons de segmenter l'image en 8 phases. Ensuite pour que l'algorithme ne tombe pas dans un minimum local il faut initialiser la température avec une valeur suffisamment grande pour que plus de la moitié des pixels de l'image soit modifiée au cours des premières itérations. Pour cela, on prendra $T^0 = 0,1$. Ensuite, il faut s'assurer que la décroissance soit très lente c'est pourquoi on choisit $\alpha = 0,99$. (On aurait pu prendre 0,9999 mais cela demande beaucoup trop d'itérations et donc beaucoup trop de temps pour que l'algorithme converge). Avec ces valeurs et au bout de 5000 itérations, l'algorithme converge et approxime le minimum global. Il reste maintenant à trouver le paramètre de régularisation optimal pour que visuellement la segmentation soit la meilleure. Nous pouvons voir sur la figure 20, des images segmentées avec différents paramètres de régularisation. Nous pouvons ainsi dire que le paramètre de régularisation donnant la meilleure segmentation est 0,012.

Nous avons vu qu'avec cette méthode, nous trouvons le minimum global de la fonction énergie. Cependant ce minimum global étant obtenu théoriquement avec un nombre infini d'itérations, il est clair qu'en pratique nous devons arrêter l'algorithme au bout d'un certain nombre d'itérations. Nous considérerons alors que nous sommes très proche du minimum global.

L'inconvénient majeur de cette méthode est le temps de calcul puisqu'il nécessite plusieurs minutes pour segmenter une image. Traiter plus d'une centaine de coupes demande donc plusieurs heures ce qui est incompatible pour une utilisation en milieu médical.

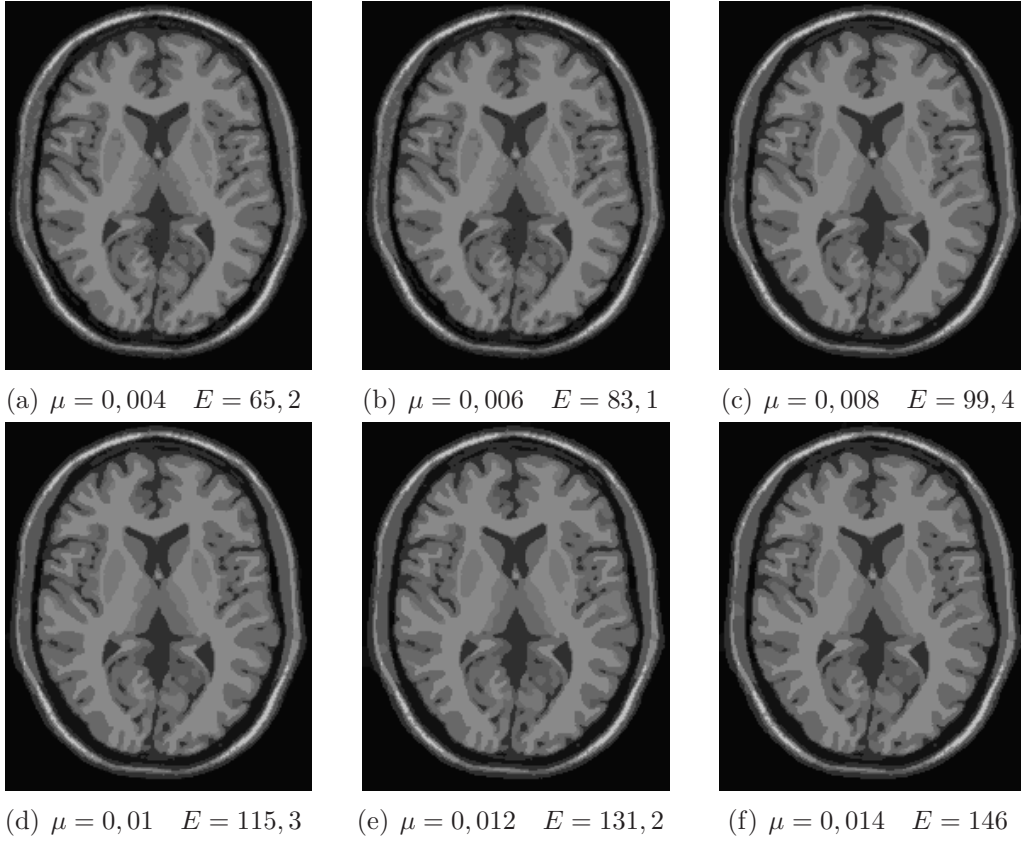


FIG. 20 – Segmentation de la coupe 80 de Brainweb par l’algorithme du recuit simulé avec plusieurs paramètres de régularisation [autres paramètres : 8 phases, $\alpha = 0.99$, 5000 itérations, $T^0 = 0, 1$].

3.3 Méthode ICM

3.3.1 Principe de l'algorithme

Comme nous avons pu le voir avec les deux méthodes précédentes, il y a un manque de rapidité. Nous allons donc implémenter une méthode plus rapide, l'ICM. Elle découle de la méthode du recuit simulé. Contrairement à cette dernière, on trouve un minimum local et non global. Au lieu de tirer au hasard une image étiquette à chaque itération, cet algorithme teste pour chaque pixel, quelle sera la phase qui minimisera l'énergie en ce pixel. Ainsi, nous sommes assurés que l'algorithme diminue l'énergie mais il y a le risque que l'algorithme s'arrête sur un minimum local car aucune remontée d'énergie est possible.

Le principe de l'algorithme est le suivant :

- On parcourt les pixels $p(i,j)$ de l'image
- Calcul de la somme des énergies d'hétérogénéité et de longueur pour chaque phase sur les pixels $p(i,j)$
- Choix de la phase qui minimise le plus l'énergie au pixel $p(i,j)$
- Mise à jour des valeurs moyennes des phases

L'algorithme s'arrête lorsque la variation d'énergie entre deux itérations est très petite ou lorsque le taux de changements est faible.

La régularisation s'effectue comme celle du recuit simulé à la différence que toutes les phases sont testées. Dans le tableau ci-dessous, sont regroupées les valeurs de l'énergie de régularisation pour chaque phase calculée à partir de la matrice $Image_{(n-1)}$ ($Image_{(n-1)}$ est celle de la page 34). On remplace successivement le pixel central de cette matrice par toutes les phases possible et on calcule la régularisation comme indiqué au chapitre 3.2.3.

Phase	1	2	3	4
Régularisation	$1 + \frac{1}{\sqrt{2}}$	$3 + \frac{2}{\sqrt{2}}$	$4 + \frac{4}{\sqrt{2}}$	$1 + \frac{1}{\sqrt{2}}$

A ces résultats sont ensuite ajoutées les énergies d'hétérogénéité pour chaque phase et enfin l'algorithme choisit la phase qui a minimisé cette somme d'énergie.

L'algorithme converge au bout d'un petit nombre d'itérations (entre 15 et 30 itérations selon les cas). Cela équivaut à un temps de calcul d'environ 10 secondes ce qui est beaucoup plus rapide que le recuit simulé.

3.3.2 Résultats

Avec cet algorithme, nous pouvons faire varier plusieurs paramètres. Le premier, comme dans les autres algorithmes, est le paramètre de régularisation, le deuxième est le nombre de phase et le dernier est l'initialisation des phases. Ce dernier ne sera pas étudié dans cet exemple. En effet, nous utilisons l'initialisation

automatique qui est étudié plus loin (section 4.1). Ensuite, pour permettre une comparaison entre les méthodes, nous choisirons d'avoir 8 phases. Il nous reste donc à effectuer une étude sur le paramètre de régularisation (Figure 21).

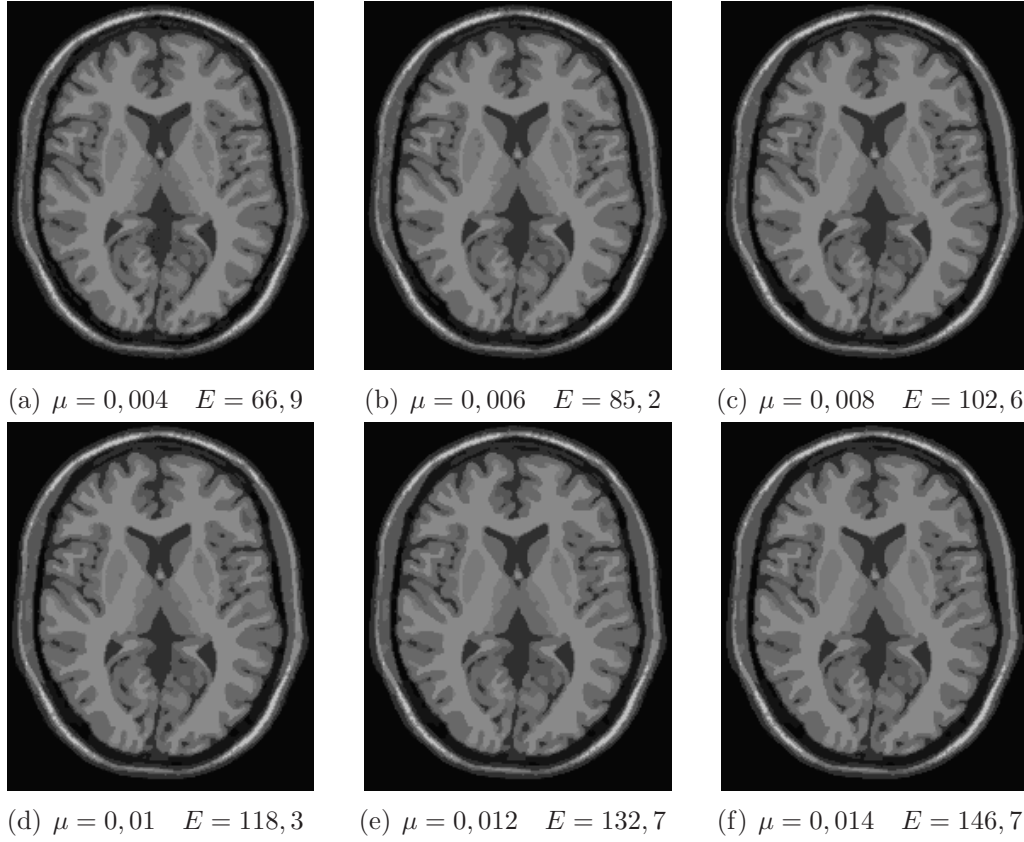


FIG. 21 – Segmentation par ICM de la coupe 80 de Brainweb selon plusieurs paramètres de régularisation

Au vu de ces résultats, le paramètre de régularisation optimal est aux alentours de 0,012 notamment parce que visuellement les phases sont plus lisses. La segmentation d'une image dure une dizaine de secondes ce qui est bien meilleur que l'algorithme du recuit simulé qui nécessite plusieurs minutes.

Nous pouvons remarquer que l'énergie obtenue avec l'ICM est très légèrement supérieure à celle obtenue avec le recuit simulé. Par exemple, pour $\mu = 0,012$, on obtient une énergie de 118,6 pour l'ICM et 115,3 avec le recuit simulé ce qui signifie que l'ICM atteint un minimum local mais il reste très proche du minimum global. Ce résultat ne confirme pas celui de la publication [13] qui montrait qu'avec l'ICM, l'énergie obtenue est 8 fois supérieure à celle obtenue avec le recuit simulé.

4 Initialisation automatique des phases

L'initialisation des phases est un élément très important de ces types d'algorithmes et une bonne initialisation permet souvent d'améliorer les résultats et aussi de réduire le nombre d'itérations nécessaire pour que les algorithmes convergent. Chan et Vese recommandaient, pour leur méthode, d'initialiser les level-set par des cylindres [14]. Concernant l'ICM, l'initialisation doit être totalement différente puisqu'il n'utilise pas les level-set. Dès les premières applications de l'ICM sur des images, je me suis rendu compte que cet algorithme était très dépendant de l'initialisation des phases et qu'il était nécessaire d'initialiser automatiquement les valeurs moyennes des phases afin de simplifier le travail de l'utilisateur. Bien évidemment, une initialisation manuelle avec une connaissance a priori de l'image permettra de mieux segmenter l'image.

4.1 Initialisation des phases en segmentation monocanal

Nous proposons, dans ce paragraphe, différentes méthodes pour initialiser les phases pour les algorithmes du recuit simulé et ICM. Toutes les méthodes données prennent en compte l'histogramme de l'image. Pour plus de clarté, tous nos exemples seront fournis pour une image 2D, le cas tridimensionnel étant une extension simple à mettre en oeuvre. Nous supposons également que nous segmentons l'image avec un nombre connu de phases (5 par exemple). En effet, nous n'avons pas cherché des méthodes permettant de déterminer automatiquement le nombre de phases nécessaire pour bien segmenter l'image (le nombre de phases varie suivant le type de recherche) mais comment déterminer la valeur moyenne des phases à partir de l'image et du nombre de phases.

Avant de détailler les méthodes, voici quelques notations qui seront utilisées dans la suite de ce paragraphe. On notera N le nombre de pixel de l'image, m le minimum d'intensité pour un pixel ($m \geq 0$), M le maximum ($M \leq 255$), H_i sera l'histogramme de l'image. Ainsi les valeurs d'intensité des pixels sont comprises entre m et M . Enfin on notera P le nombre de phases. Pour les exemples numériques, nous prendrons $m = 0$ et $M = 255$.

4.1.1 Première méthode

Une première méthode consiste tout simplement à diviser la plage de valeur de l'image $[m M]$ en P segments égaux, puis à prendre le milieu de chaque segment pour valeur moyenne de chaque phase. Par exemple, si nous divisons le segment $[0 255]$ en 5 segments nous obtiendrons les valeurs d'initialisation suivantes $[25 75 125 175 225]$. On peut immédiatement dire que cette initialisation n'est pas optimale puisque les structures que nous voulons séparer (matière grise, blanche...) ne sont pas équiréparties sur la plage $[m M]$.

4.1.2 Deuxième méthode

Une deuxième méthode consiste à diviser la plage $[mM]$ en K segments mais en s'assurant que chaque segment ait le même nombre de pixel c'est à dire $\frac{N}{P}$ pixels. Nous avons choisi de représenter les segments par trois manières différentes. On peut initialiser les valeurs moyennes aux extrémités supérieures, inférieures ou au milieu des segments. Si, par exemple, les intervalles sont les suivants : $[0\ 5]$, $[6\ 30]$, $[31\ 110]$, $[111\ 200]$ et $[201\ 255]$ alors une initialisation aux bornes inférieures donne $[0\ 6\ 31\ 11\ 201]$, une initialisation aux bornes supérieures donne $[5\ 30\ 110\ 200\ 255]$ et une initialisation au milieu des intervalles donne $[2,5\ 18\ 70,5\ 155,5\ 233]$. On peut voir sur les 6 images de la figure 22 ce qu'ont donné les trois initialisations et le résultat de la segmentation.

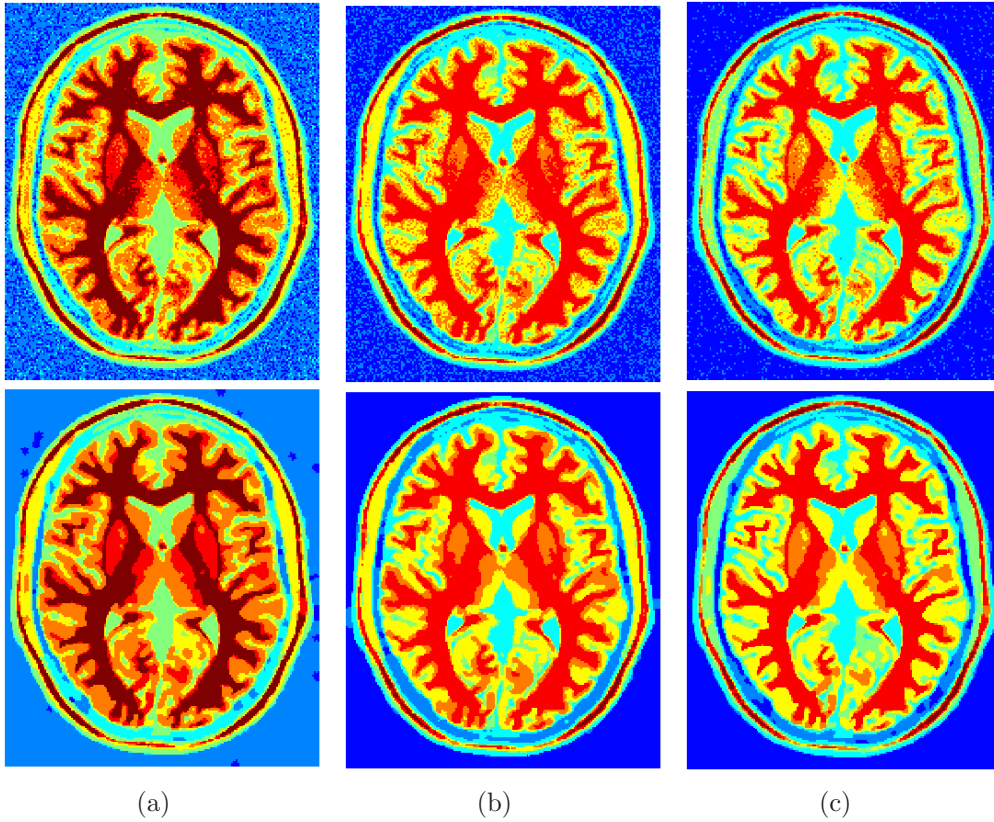


FIG. 22 – Première ligne : carte de l'initialisation des phases selon les 3 méthodes (bornes inférieures (a), milieu des intervalles (b) et bornes supérieures (c)). Deuxième ligne : résultat de la segmentation avec les initialisations faites au dessus.

Il est très difficile de privilégier une méthode par rapport à une autre. En effet, l'initialisation (a) permet d'obtenir les noyaux en entier contrairement à (b) mais la matière blanche est dans la même phase que l'os autour, ce qui n'est pas le cas dans (b) où la matière blanche et l'os sont dans deux phases distinctes. Cependant, on pourra choisir l'initialisation au milieu des intervalles qui paraît la méthode la plus logique.

4.2 Initialisation des phases en multicanal

Pour une segmentation multicanal, l'initialisation devient plus complexe puisque l'information est différente sur les deux canaux. Il se pose plusieurs problèmes. Le premier concerne les valeurs moyennes des phases sur chaque canal car la matière blanche par exemple n'a pas la même valeur moyenne que l'on soit sur le canal 1 ou 2. Le deuxième problème concerne l'attribution des pixels à une étiquette car une structure n'aura pas les mêmes pixels sur le canal 1 et sur le canal 2.

Deux méthodes ont été envisagées pour initialiser les phases mais finalement une seule s'est révélée viable. Nous allons cependant détailler les deux méthodes et dire pourquoi nous n'en avons retenu qu'une seule.

4.2.1 Première méthode

La première méthode, celle qui a été retenue, est la plus simple. Elle consiste à initialiser les phases à partir d'un canal (on prendra pour cela la méthode d'initialisation donnée au paragraphe 4.1.2) et d'obtenir la carte des étiquettes pour ce canal. On applique ensuite cette carte sur l'autre canal afin de calculer les valeurs moyennes des phases sur cet autre canal. Algorithmiquement, voici les étapes.

- Initialisation de la carte des étiquettes P_{init1} , les pixels de P_{init1} allant de 1 à P , et des valeurs moyennes $(C1(k), k = 1 : P)$ à partir de l'histogramme du canal 1.
- On applique la carte P_{init1} sur le canal 2 pour calculer les valeurs moyennes $(C2(k), k = 1 : PH)$.
- Une phase k a pour valeur moyenne $C1(k)$ sur le canal 1 et $C2(k)$ sur le canal 2.

Ainsi, nous n'avons pas de problème de fusion à l'initialisation puisque nous initialisons les phases avec l'histogramme du canal 1 et nous calculons facilement les valeurs moyennes qu'ont ces phases dans le canal 2. Voici une illustration de cette méthode (Figure 23).

4.2.2 Deuxième méthode

L'inconvénient de la méthode précédente est qu'elle favorise un canal par rapport à un autre. Nous pouvons penser que le mieux serait d'utiliser également les données des deux canaux. Voici la description d'une telle méthode, un schéma figure 24 décrivant l'algorithme permettra de mieux le comprendre. On notera que C_{ij} représente les valeurs moyennes des phases du canal i calculées à partir de l'histogramme du canal j .

- Initialisation de la carte des étiquettes P_{init1} et des valeurs moyennes des phases C_{11} du canal 1 à partir de l'histogramme du canal 1.
- On applique P_{init1} sur le canal 2 pour calculer les valeurs moyennes C_{21} .

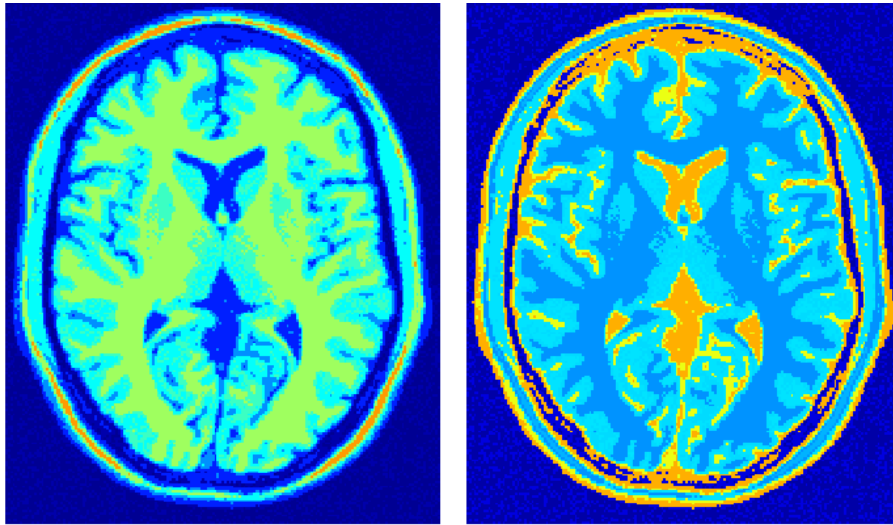


FIG. 23 – Initialisation des phases à partir de l’histogramme du canal 1. A gauche : initialisation du canal 1. A droite : initialisation du canal 2.

- Initialisation de la carte des étiquettes P_{init2} et des valeurs moyennes des phases $C22$ du canal 2 à partir de l’histogramme du canal 2.
- On suppose que les valeurs moyennes $C21$ et $C22$ sont quasiment identiques mais que l’ordre est changé. Nous trions $C21$ pour mettre en correspondance $C11$ et $C22$.

Au vu des résultats de la figure 25, l’initialisation est moins bonne que la méthode précédente comme on peut le voir sur l’image en bas à droite. Cela est dû au fait que la prise en compte des valeurs de l’initialisation du canal 2 “pollue” l’initialisation du canal 1 lors de la fusion. De plus, un inconvénient de cette méthode vient du fait de l’hypothèse qui a été établie sur l’ordre des valeurs moyennes $C21$ et $C22$. les images étant parfois peu contrastées (pour les images en mode T2, la matière blanche a la même intensité que la matière grise), il est possible que l’ordre soit modifié ce qui entraîne une fausse correspondance entre valeurs moyennes de phase et donc une mauvaise segmentation.

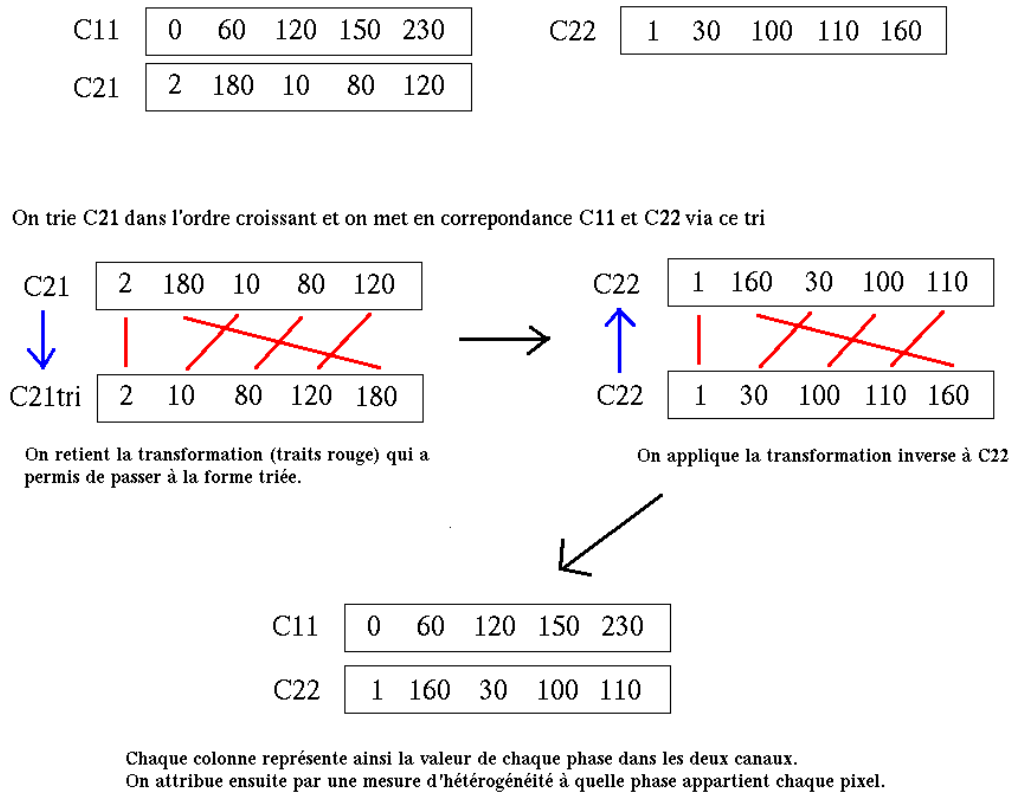


FIG. 24 – Schéma décrivant la méthode 2 d'initialisation multicanal. On rappelle que C_{ij} sont les valeurs moyennes des phases du canal i calculées à partir de l'histogramme du canal j . Supposons qu'on ait C11, C21 et C22. Pour mettre en correspondance les valeurs moyennes des phases du canal 1 (C11) et du canal 2 (C22), on utilise la transformation qu'il faut effectuer pour trier C21 dans l'ordre croissant.

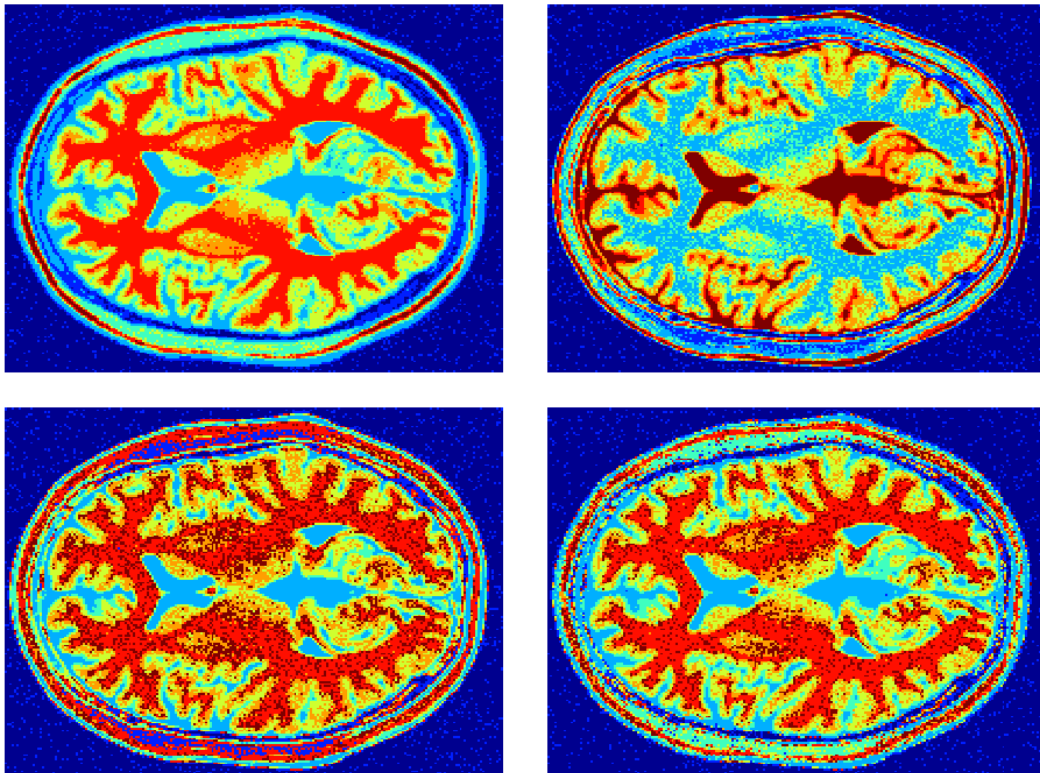


FIG. 25 – L'image en haut à gauche est *Pinit1* et celle en haut à droite est *Pinit2*. La mise en correspondance des phases s'effectue au moyen de *C21* et on recalcule les labels des phases (en bas à gauche) comme le montre le changement de couleurs. On fusionne ensuite les deux images de gauche ce qui donne la carte d'initialisation finale (en bas à droite). (coupe 80 de Brainweb)

5 Application de la segmentation sur des images réelles

Il existe de multiples applications de segmentation d'images cérébrales. Nous allons segmenter des images d'IBSR afin d'obtenir des phases représentant le fond, la matière blanche, la matière grise ainsi que le LCR. Puis nous segmenterons des images avec une pathologie afin de montrer l'avantage du multicanal. Toutes les segmentations seront réalisées avec l'ICM puisque nous avons montré son efficacité. Nous comparerons les résultats de la segmentation avec des segmentations manuelles. Pour cela nous avons besoin de critères de comparaison.

5.1 Critères de comparaison

La comparaison entre la segmentation automatique et la segmentation manuelle sera faite avec les 4 indices suivants, le pourcentage de vrai positif, faux positif, de l'indice de Jaccard et de similarité entre la segmentation manuelle et automatique.

Commençons tout d'abord par les deux premiers. Les vrais positifs (notés VP) sont les pixels qui sont communs aux deux segmentations, les faux positifs (notés FP) sont les pixels qui sont dans la segmentation ICM mais pas dans la segmentation manuelle. Le schéma ci-dessous permet de visualiser ces deux termes.

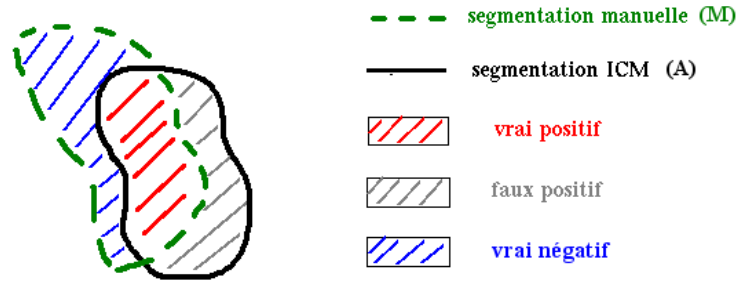


FIG. 26 – Schéma représentant les vrais positifs, faux positifs et faux négatifs

Nous pouvons définir mathématiquement ces deux termes. Soit A , la tumeur de la segmentation ICM et M la tumeur de la segmentation manuelle. On note $|X|$ le nombre de pixels de X . Le pourcentage de vrai positif et de faux positif est donné par les formules suivantes.

$$VP = \frac{|A \cap M|}{|M|} \quad (19)$$

$$FP = \frac{|A| - |A \cap M|}{|A|} = 1 - \frac{|A \cap M|}{|A|} \quad (20)$$

Définissons maintenant les indices de similarité (Sim) et de Jaccard (Jac) :

$$Sim = 2 \frac{|A \cap M|}{|A| + |M|} \quad (21)$$

$$Jac = \frac{|A \cap M|}{|A \cup M|} \quad (22)$$

Ces indices permettent de prendre en compte conjointement les vrais positifs et les faux positifs. On remarquera que si $A = M$, c'est à dire si la segmentation ICM et la segmentation manuelle sont identiques alors $VP = 1$, $FP = 0$, $Sim = 1$, et $Jac = 1$. Ainsi plus ces indices se rapprocheront de ces valeurs, meilleures seront les segmentations. Nous nous baserons principalement sur ces 4 nombres pour juger de la qualité de la segmentation.

Il est possible de calculer les indices de Jaccard et de similarité à partir des vrais et faux positifs comme démontré dans l'annexe A). On obtient les formules suivantes :

$$\frac{1}{Jac} = \frac{1}{VP} + \frac{1}{1 - FP} - 1 \quad (23)$$

$$\frac{1}{Sim} = 0,5 \left(\frac{1}{VP} + \frac{1}{1 - FP} \right) \quad (24)$$

De plus, nous pouvons également calculer ces indices l'un à partir de l'autre

$$Jac = \frac{Sim}{2 - Sim} \quad (25)$$

$$Sim = 2 \frac{Jac}{Jac + 1} \quad (26)$$

Pour améliorer cette qualité, il sera nécessaire d'effectuer des opérations de prétraitement et de post-traitement qui seront détaillées pour chaque cas. Dans un premier temps, nous segmenterons un volume d'un cerveau normal, c'est à dire n'ayant aucune anomalie. Le but sera de segmenter l'image pour obtenir une phase pour le fond, une phase pour le LCR, une phase pour la matière grise et enfin une dernière pour la matière blanche. Ensuite nous segmenterons un volume ayant une tumeur. Ces images seront des images SPGR et FLAIR en monocanal puis nous montrerons l'avantage d'une segmentation multicanal combinant ces deux données.

5.2 Segmentation d'un cerveau en 3 phases : matière grise, matière blanche et LCR

5.2.1 Prétraitement des images

Pour améliorer la segmentation, il faut traiter l'image avant qu'elle ne soit segmentée. L'inconvénient des images IBSR est que la segmentation manuelle associée ne prend pas en compte le LCR qui entoure le cerveau mais uniquement celui qui se situe dans les ventricules. On masque donc d'abord le volume par la segmentation manuelle afin de n'obtenir que le volume qui a été réellement segmenté manuellement. Bien entendu, cette opération n'a qu'une valeur de comparaison car, dans la réalité et pour une application dans le milieu médical, il n'y aurait pas de segmentation manuelle puisque nous cherchons justement à remplacer la segmentation manuelle par des méthodes automatiques.

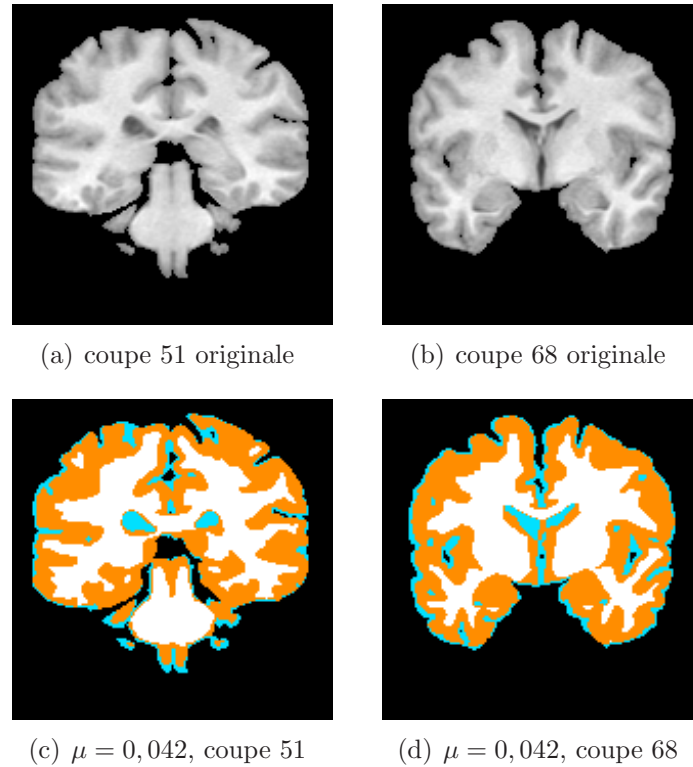


FIG. 27 – Segmentation de deux coupes de IBSR 5 en 4 phases : le fond (noir), la matière grise (orange), la matière blanche (blanc) et le LCR (bleu clair)

5.2.2 Résultat bruts

J’ai segmenté un volume de IBSR avec 4 phases. Voici les résultats obtenus (figure 27) immédiatement après la segmentation sans post-traitement.

Nous pouvons faire les remarques suivantes :

- Du LCR est trouvé autour du cerveau et dans les sillons de la matière grise alors que ces zones doivent être de la matière grise.
- La matière blanche est bien segmentée.

Pour régler ces problèmes, il faut effectuer des opérations post-segmentation et c’est ce qui est détaillé dans la suite.

5.2.3 Post-traitements

Nous avons dit précédemment que le LCR ne devait pas être pris en compte lorsque celui-ci est autour de la matière grise, il est donc nécessaire de faire des opérations post-segmentation afin d’améliorer les résultats. Voici la chaîne de traitement qui a été mise en place.

Étape 1 Une première opération consiste à “coller” la matière blanche sur le LCR des ventricules. Il suffit de supprimer la matière grise qui se situe entre le LCR et la matière blanche. Pour cela on convertira en matière blanche, les zones de

matière grise qui ont des aires inférieures à un certain seuil (30 pixels en volume) et qui sont connexes avec les ventricules.

Etape 2 On effectue une opération de dilatation de la matière grise sur le LCR. En effet, nous voulons convertir celui-ci en matière grise mais cette opération de dilatation ne permet uniquement de dilater que d'une couche la matière grise. Cela permet de supprimer les petites zones de LCR et aussi de couper le contact entre le LCR et l'extérieur. Cela permet parfois de conserver comme nous le verrons plus tard des zones de LCR qui doivent être segmentées en LCR et non convertit en matière grise.

Etape 3 On convertit en matière grise tout le LCR qui est en contact avec l'extérieur d'où l'utilité de l'étape précédente.

Etape 4 On dilate le LCR sur la matière grise. C'est l'opération inverse de l'étape 2. Cela permet de retrouver les pixels de LCR perdus lors de cette étape 2.

Etape 5 En partant de la remarque que le LCR est toujours en contact avec de la matière blanche (d'où l'utilité de l'étape 1), on supprime toutes les zones de LCR qui sont entourées uniquement de matière grise.

Après toutes ces opérations de post-traitement, nous obtenons un cerveau segmenté proche de celui de la segmentation manuelle (figure 28). Une étude paramétrique a ensuite été réalisée (voir courbes en annexe B). Les meilleurs résultats sont obtenus avec une régularisation de 0,042. Nous pouvons faire une remarque sur la segmentation des noyaux qui, comme vous pouvez le voir sur la figure 28, ne sont pas segmentés. Ceci est dû au fait que les noyaux sont très peu contrastés sur l'image originale et qu'il faut l'œil du médecin pour extraire les noyaux. Cela influe donc beaucoup sur la valeur des indices.

5.3 Segmentation des tumeurs

5.3.1 Images traitées

Dans ce paragraphe, nous allons segmenter des tumeurs. Nous allons donner les résultats en segmentant en monocanal et en multicanal des images SPGR et FLAIR. Nous donnerons le pourcentage de vrai positif et de faux positif, l'indice de similarité et de Jaccard ainsi que la distance moyenne et maximale. Les segmentations ont été faites avec 7 phases sur les images SPGR en monocanal, 5 phases sur les images FLAIR en monocanal et 7 phases en multicanal. Les principaux résultats sont consignés dans le tableau (figure 29 et nous commentons ces résultats dans les paragraphes qui suivent. La totalité des études paramétriques sur la segmentation de la tumeur du cas 1 sont dans les annexes C, D et E.

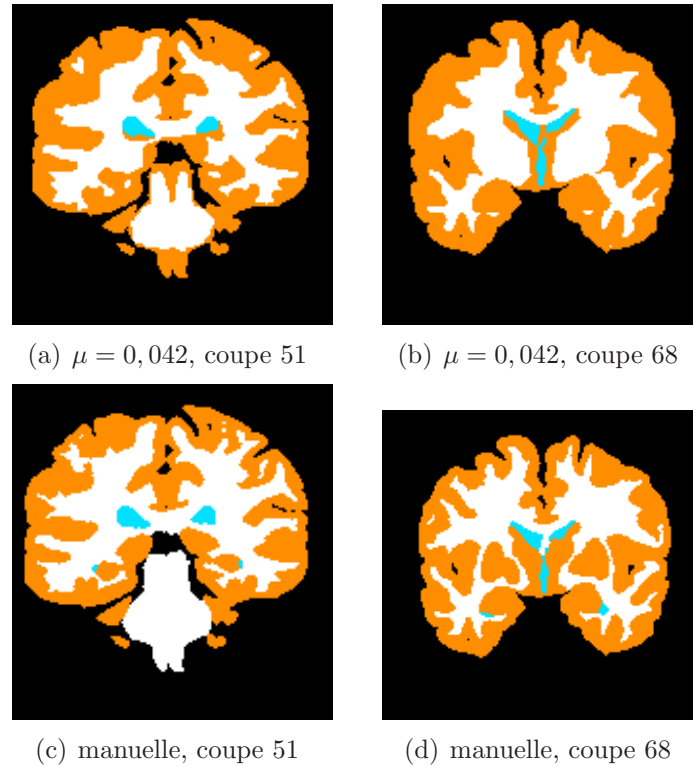


FIG. 28 – Segmentation en 4 phases après post-traitement (a) et (b) de deux coupes de IBSR à comparer avec les segmentations manuelles (c) et (d)

5.3.2 Segmentation de l'image SPGR

Comme nous pouvons nous y attendre, les résultats de la segmentation de la tumeur sur une image SPGR sont mauvais mais ce type d'image n'est pas utilisé pour voir les tumeurs. Vous pouvez voir les contours de la tumeur qui ont été obtenus manuellement (en vert) et par la méthode ICM (en rouge). Sur le volume, on atteint 72% de vrai positif et 21% de faux positif.

5.3.3 Segmentation de l'image FLAIR

L'image FLAIR donne un excellent contraste pour la tumeur et donne donc de très bons résultats. On atteint 81% de vrai positif pour le cas 1 et 76% pour le cas 2. Le pourcentage de faux positif est aussi excellent puisque nous avons seulement 0,6%.

Une étude portant sur la paramètre de régularisation a été faite dont les résultats sont consignés dans le tableau en annexe. Nous pouvons voir que les résultats sont globalement bons quelque soit le paramètre de régularisation. On atteint même 82% de vrai positif et 10% de faux positif. Les indices sont également bons (0,85 pour la similarité et 0,75 pour l'indice de Jaccard). Ces résultats sont illustrés sur la figure 5.3.3 donnant les contours des segmentations automatiques et manuelles superposés à l'image originale.

	VP	FP	Sim	Jac
Cas 1				
Mo SPGR/SPGR	72	21	75	60
Mo FLAIR/FLAIR	81	9,5	85	75
Mu /SPGR	76	27	74	59
Mu /FLAIR	77	6	84	73
Cas 2				
Mo SPGR/SPGR	38	77	28	16
Mo FLAIR/FLAIR	76	0,5	86	75
Mu /SPGR	68	1,45	80	67
Mu /FLAIR	80,7	2,6	88	79

FIG. 29 – Evaluation des vrais positifs, faux positifs, indices de similarité et de Jaccard (en pourcentage) pour deux cas de tumeurs sur des segmentations monocanal (Mo) et multicanal (Mu) comparé aux segmentations manuelles associées.

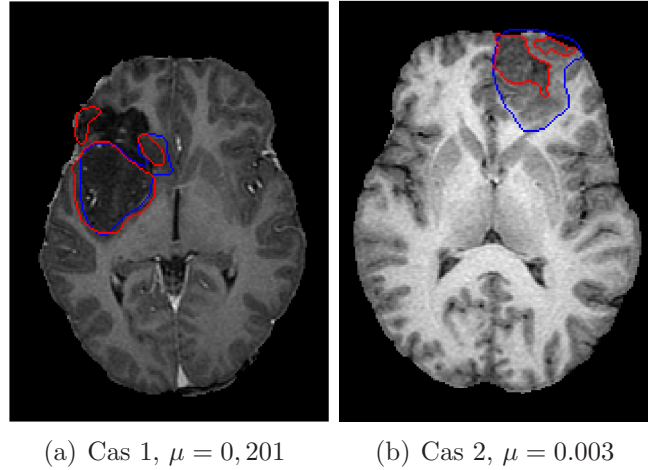


FIG. 30 – Contours des tumeurs par segmentation ICM (rouge) et manuelle (en bleu) sur SPGR.

5.3.4 Segmentation multicanal des images SPGR-FLAIR

La segmentation multicanal utilise la complémentarité des images SPGR et FLAIR pour détecter la tumeur et également la matière grise, la matière blanche et le LCR. Une question porterait sur la segmentation manuelle à utiliser pour la comparaison. Nous avons choisi de comparer les résultats de la segmentation multicanal à la segmentation manuelle FLAIR et SPGR tout en gardant à l'esprit que la segmentation manuelle FLAIR est la plus proche de la réalité.

Les résultats de l'étude paramétrique portant sur le paramètre de régularisation sont en annexe E. Le pourcentage de vrai positif (77% pour le cas 1 et 80% pour le cas 2) montre que la segmentation de la tumeur en multicanal se rapproche de la segmentation FLAIR comme illustrée sur la figure 5.3.4.

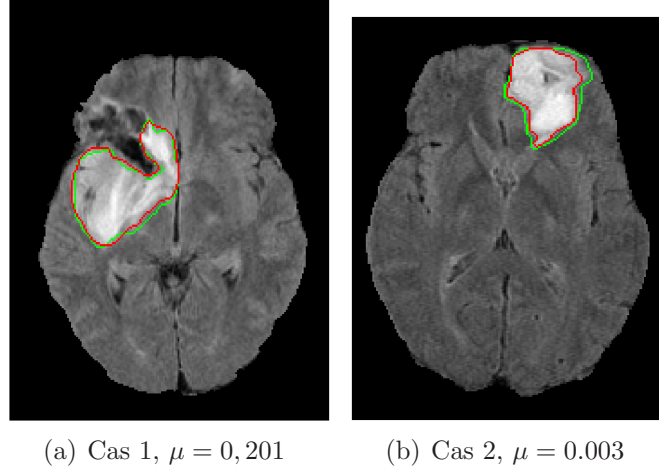


FIG. 31 – Contours des tumeurs par segmentation ICM (rouge) et manuelle (en vert) sur FLAIR

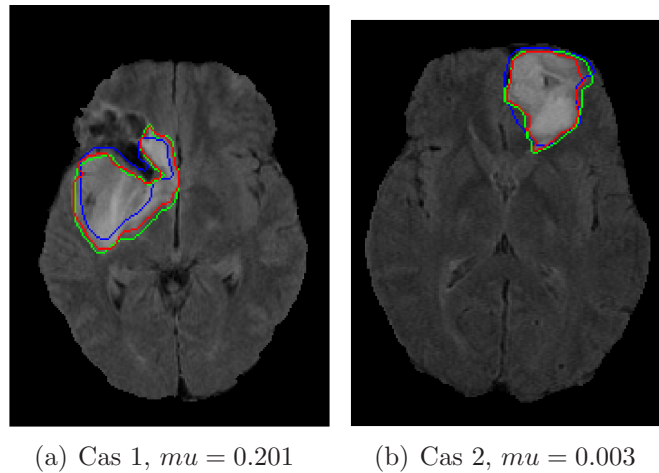
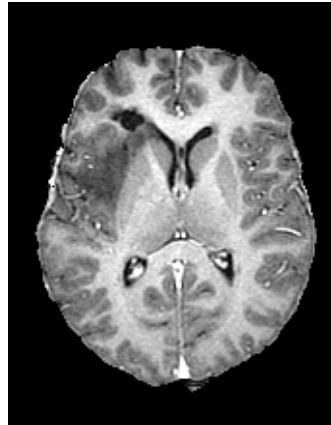


FIG. 32 – Contours des tumeurs par segmentation multicanal (rouge) et manuelle (vert pour FLAIR et bleu pour SPGR).

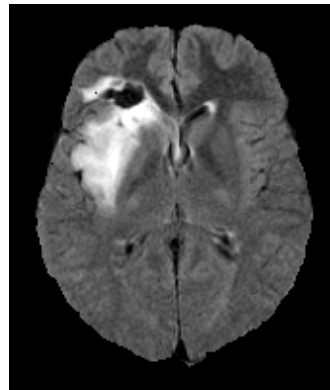
5.4 Segmentation d'un cerveau entier avec une tumeur

Le paragraphe précédent avait pour but de montrer que nous pouvons segmenter spécifiquement une tumeur. Mais le but d'une segmentation multicanal est de pouvoir fusionner les données des deux types d'image (SPGR et FLAIR) et ainsi d'obtenir sur une même image les parties saines et les zones malades (figure 33). Un problème réside dans le fait que la fusion ne privilégie pas le canal SPGR pour les parties saines et le canal FLAIR pour la tumeur. On peut voir sur la figure 33(c), que la matière grise est mal segmentée à certains endroits à cause du canal FLAIR. Une solution pour sensiblement améliorer les résultats tout en gardant une bonne segmentation de la tumeur est d'accorder un poids plus important à l'image SPGR. Comme la tumeur est très contrastée sur l'image FLAIR, cela ne modifie pas la segmentation de celle-ci. Le poids est inséré lors du calcul d'hétérogénéité. On multiplie le facteur de normalisation du canal SPGR par un coefficient. Plus ce poids est important, plus la segmentation se rapproche d'une segmentation monocanal du canal SPGR. Un poids de 5 est suffisant pour mieux segmenter les

structures saines tout en gardant la segmentation de la tumeur.



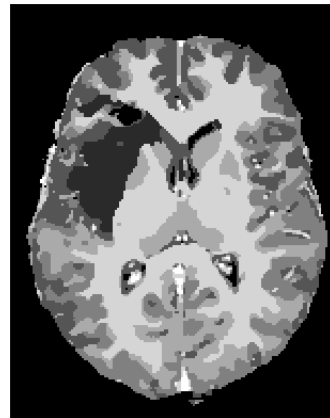
(a) coupe SPGR originale du cas 1



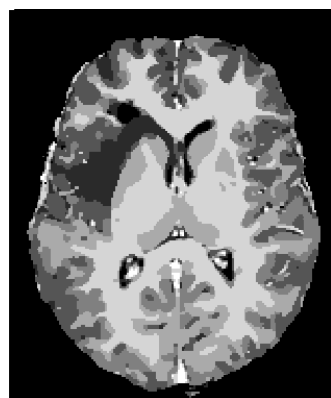
(b) coupe FLAIR originale du cas 1



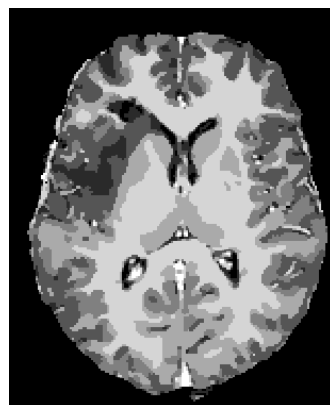
(c) poids 1 (le poids des deux canaux sont égaux)



(d) poids 5



(e) poids 20



(f) poids 100

FIG. 33 – Segmentation avec un poids plus fort pour le canal SPGR

Conclusion

Nous avons montré que la méthode ICM permettant de résoudre le problème de minimisation d'énergie de Mumford-Shah est tout à fait adaptée pour traiter des images médicales. Elle permet non seulement de traiter une image plus rapidement que la méthode de Chan et Vese utilisant les level-set mais aussi de converger vers une solution proche de celle de l'algorithme du recuit simulé censée être la solution optimale. De plus, la facilité d'utilisation et la simplicité des calculs en font un algorithme aisément utilisable par des médecins.

La grande adaptabilité de l'ICM, notamment en terme du choix de nombre de phases, permet de rechercher dans l'image plusieurs types de structures. Nous avons vu qu'une segmentation à 4 phases permettait d'obtenir la matière blanche, la matière grise et le LCR. En augmentant le nombre de phases, nous pouvons extraire du cerveau des petites structures comme les noyaux. La recherche de tumeurs est également possible sur les images FLAIR avec un nombre limité de phases. Malgré la non spécificité de cet algorithme dans la segmentation des tumeurs et en comparaison avec une segmentation manuelle, nous atteignons 81% de vrais positifs et nous descendons à 0,6% de faux positifs ce qui en fait une méthode fiable de segmentation de tumeur.

Ensuite, l'ICM est applicable pour un traitement multicanal permettant ainsi d'utiliser conjointement les données des acquisitions SPGR et FLAIR et d'obtenir sur une même image les structures saines de l'image SPGR et les zones malades de l'image FLAIR. Ces résultats s'améliorent si un poids plus fort est donné à l'image SPGR.

Il semble pertinent de poursuivre des recherches dans cette voie. Il pourrait être judicieux de prendre en compte d'autres informations, comme les informations spatiales, afin de contraindre l'évolution des phases. Ce stage s'étant très largement intéressé aux applications sur les images de type SPGR et FLAIR, il pourrait être utile de l'appliquer sur d'autres modalités d'imagerie comme le scanner ou la scintigraphie notamment concernant la segmentation multicanal.

Références

- [1] P. Brigger, J. Hoeg, M. Unser, "B-spline snakes : a flexible tool for parametric contour detection," IEEE Trans. Image Processing, vol.9, no.9, Sept 2000.
- [2] T. F. Chan, L. A. Vese, "Active contour without edges," IEEE Trans. Image Processing, vol. 10, no. 2, Feb 2001.
- [3] T. F. Chan, B. Y. Sandberg, L. A. Vese, "Active contour without edges for vector-valued images," J. Visual Commun. Image Representation 11, 130-141, 2000.
- [4] J. Chung, L. A. Vese, "Image segmentation using a multilayer level-set approach," UCLA C.A.M Report 03-53, 2003
- [5] L. D. Cohen and I. Cohen, "Finite-element method for active contour models and balloons for 2D and 3D Images," IEEE Trans. Pattern Anal. Machine Intell., vol. 15, pp. 1131-1147, Nov 1993.
- [6] D. Cremers, N. Sochen, C. Schnörr, "Towards recognition based variational segmentation using shape priors and dynamic labeling." Scale-Space, pp. 388-400, 2003
- [7] D. Cremers, N. Sochen, C. Schnörr, "A multiphase dynamic labelling model for variational recognition-driven image segmentation", Int. J. Comput. Vis., pp 67-81, 2006.
- [8] V. Israel-Jost, E. Breton, E. Angelini, P. Choquet, I. Bloch, A. Constantinesco, "Vectorial Multi-phase Mouse Brain Tumor Segmentation in T1-T2 MRI," ISBI 2008.
- [9] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes : Active contour Models," Int. J. Comput. Vis, pp. 321-331, 1988.
- [10] H. Khotanlou. 3D brain tumors and internal brain structures segmentation in MR images. PhD thesis, Ecole Nationale Supérieure des Télécommunications (telecom ParisTech), 2008.
- [11] D. Mumford, Shah, "Optimal Approximation by piecewise smooth functions and associated variational problems," Commun. Pur. Appl. Math., 42, pp. 577-685, 1989.
- [12] B. Sandberg, T. F. Chan, "A logic framework for active contours on multi-channel images," J. Vis. Commun. Image R., vol. 16, pp. 333-358, 2005.
- [13] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, C. Rother, "A Comparative Study of Energy Minimization Methods for Markov random Fields with Smoothness-Based Priors", IEEE Trans. Pattern Anal. Machine Intell., vol. 30, No. 6, pp. 1068-1080, June 2008.
- [14] L. A. Vese, T. F. Chan, "A multiphase level-set framework for image segmentation using the Mumford and Shah model," Int. J. Comput. Vis, vol.50, No. 3, pp. 271-293, 2002.
- [15] C. Xu and J. L. Prince, "Snakes, shapes, and gradient vector flow," IEEE Trans. Image Processing, vol. 7, no.3, March 1998.

Annexes

A Relation mathématique entre les indices de similarité et de Jaccard

Partons des formules 21 et 22 et décomposons les inverses des indices de similarité et de Jaccard.

$$\begin{aligned} Jac &= \frac{|A \cap M|}{|A \cup M|} \\ &= \frac{|A \cap M|}{|M| + |A| - |A \cap M|} \quad \Rightarrow \quad \frac{1}{Jac} = \frac{|M|}{|A \cap M|} + \frac{|A|}{|A \cap M|} - 1 \\ Sim &= 2 \frac{|A \cap M|}{|A| + |M|} \quad \Rightarrow \quad \frac{2}{Sim} = \frac{|A|}{|A \cap M|} + \frac{|M|}{|A \cap M|} \end{aligned}$$

En utilisant les formules de vrai et faux positifs, nous avons les deux égalités suivantes

$$\frac{|M|}{|A \cap M|} = \frac{1}{VP} \quad \text{et} \quad \frac{|A|}{|A \cap M|} = \frac{1}{1 - FP}$$

On obtient ainsi les deux formules ci-dessous permettant de calculer l'indice de Jaccard et de similarité à partir des vrai et faux positifs.

$$\begin{aligned} \frac{1}{Jac} &= \frac{1}{VP} + \frac{1}{1 - FP} - 1 \\ \frac{1}{Sim} &= 0,5 \left(\frac{1}{VP} + \frac{1}{1 - FP} \right) \end{aligned}$$

Enfin on obtient facilement la relation entre l'indice de Similarité et l'indice de Jaccard :

$$\begin{aligned} Jac &= \frac{Sim}{2 - Sim} \\ Sim &= 2 \frac{Jac}{Jac + 1} \end{aligned}$$

B Etudes paramétriques de la segmentation sur IBSR

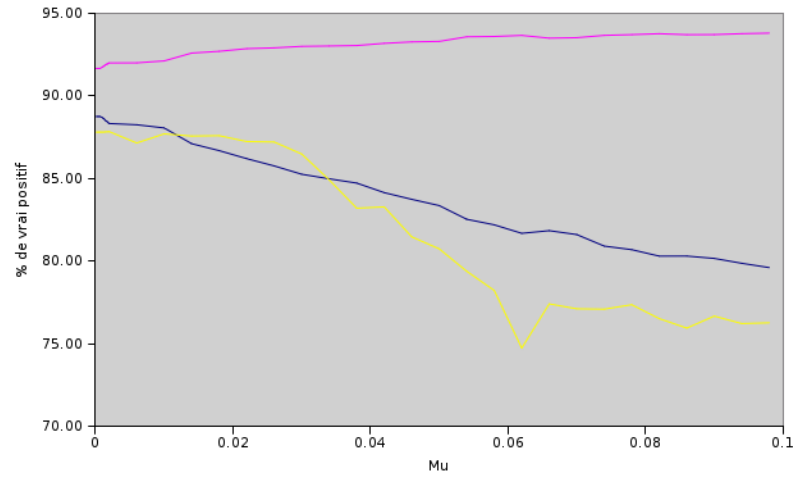


FIG. 34 – Courbes des vrais positifs de la matière blanche (bleu), de la matière grise (rose) et du LCR (jaune) en fonction du paramètre de régularisation.

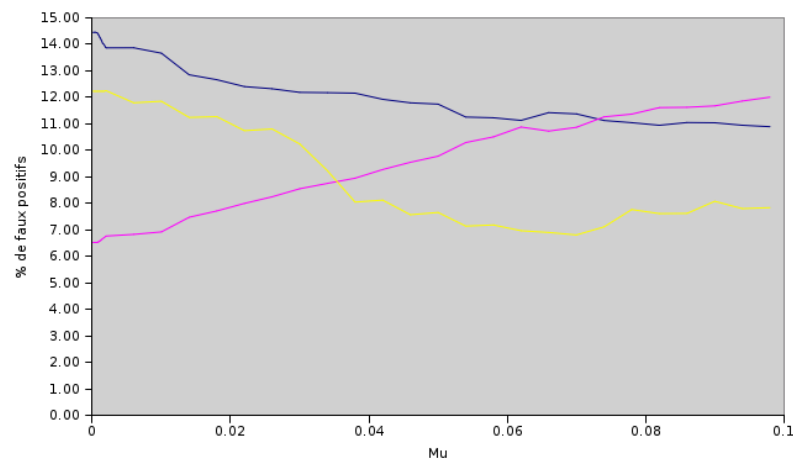


FIG. 35 – Courbes des pourcentages de faux positifs de la matière blanche (bleu), de la matière grise (rose) et du LCR (jaune) en fonction du paramètre de régularisation.

C Etude paramétrique de la segmentation de la tumeur du cas 1 en SPGR

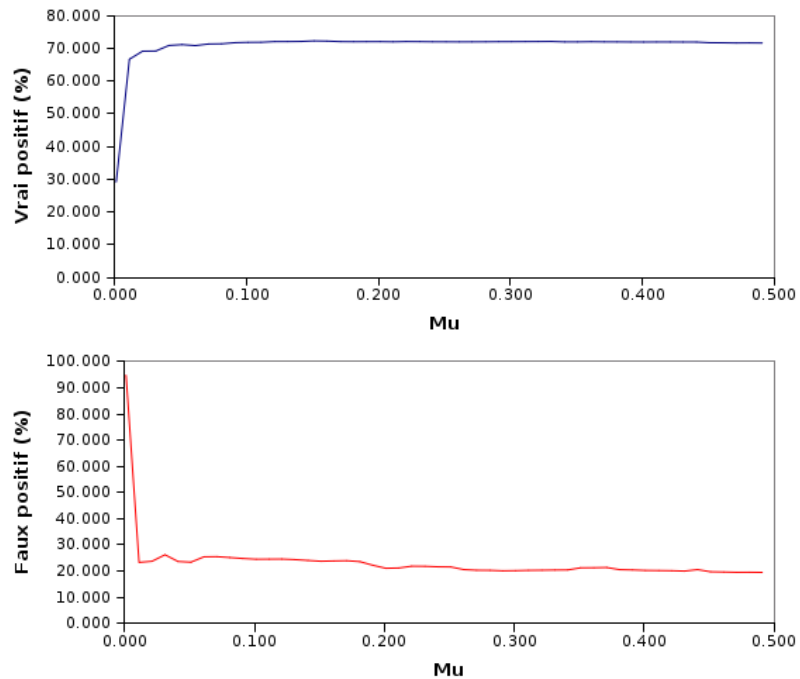


FIG. 36 – Courbes des pourcentages de vrais (bleu) et faux (rose) positifs de la segmentation de la tumeur sur le canal SPGR en fonction du paramètre de régularisation.

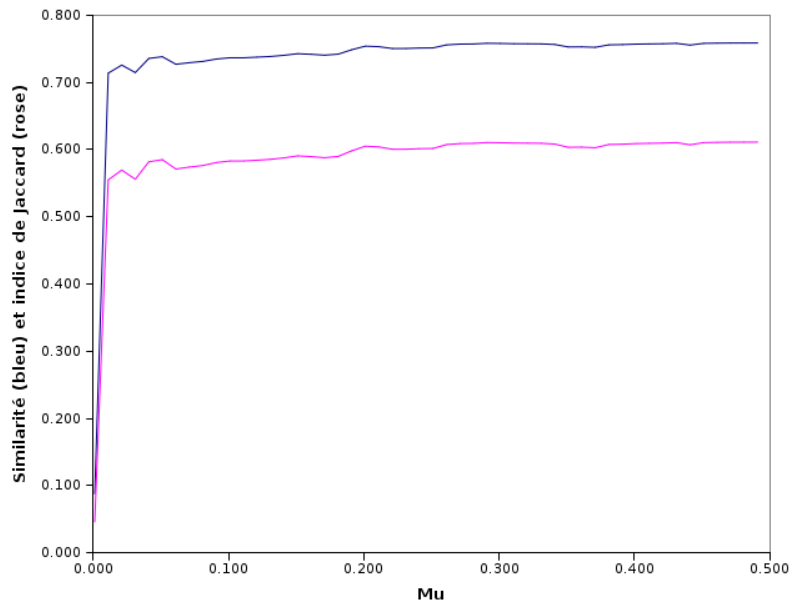


FIG. 37 – Courbes des indices de similarité (bleu) et de Jaccard (rose) de la segmentation ICM de la tumeur sur l'image SPGR du cas 1 en fonction du paramètre de régularisation.

D Etude paramétrique de la segmentation de la tumeur

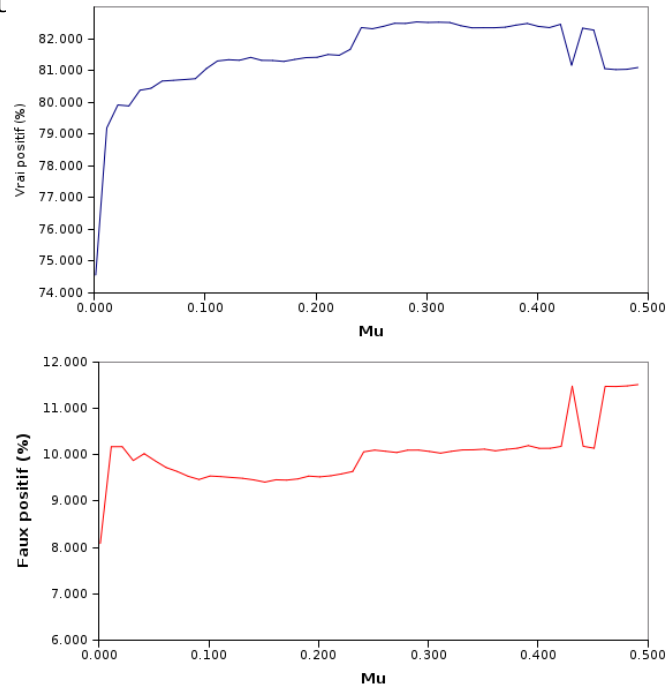


FIG. 38 – Courbes des pourcentages de vrais (bleu) et faux (rose) positifs de la segmentation de la tumeur sur le canal FLAIR en fonction du paramètre de régularisation.

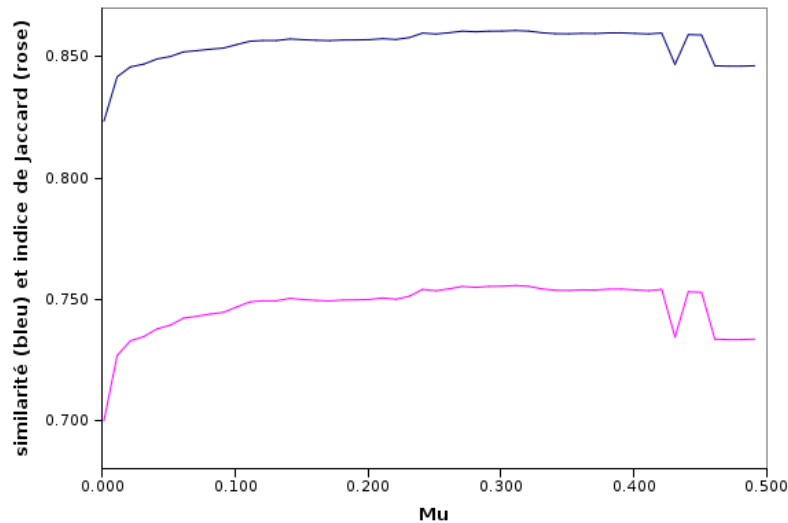


FIG. 39 – Courbes des indices de similarité (bleu) et de Jaccard (rose) de la segmentation ICM de la tumeur sur l'image FLAIR du cas 1 en fonction du paramètre de régularisation.

E Etude paramétrique de la segmentation multicanal de la tumeur du cas 1

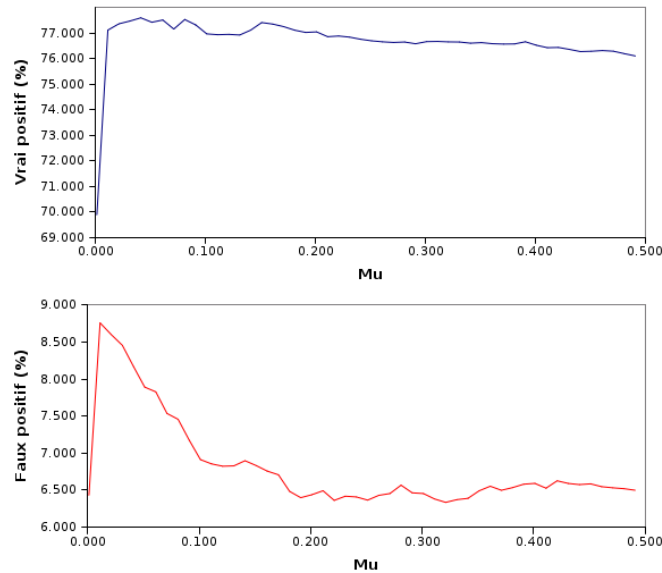


FIG. 40 – Courbes des pourcentages de vrais (bleu) et faux (rose) positifs de la segmentation multicanal de la tumeur en fonction du paramètre de régularisation.

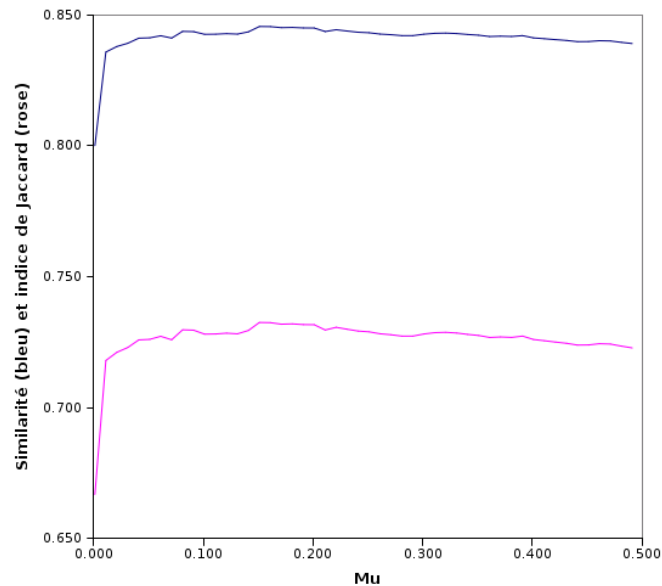


FIG. 41 – Courbes des indices de similarité (bleu) et de Jaccard (rose) de la segmentation ICM multicanal de la tumeur en fonction du paramètre de régularisation.

F Manuel d'utilisation des codes Matlab

Segmentation monocal

Voici la structure classique d'un code Matlab permettant de segmenter un volume. (on suppose que la taille du volume à segmenter est $M \times N \times P$ et on affichera la taille de chaque matrice entre parenthèses)

Lecture du fichier *nom du fichier* pour obtenir le volume *V1*.

```
[V1 HeaderI]=read_volume_img(nom du fichier,'b');
```

Avec les variables *Slice_start* et *Block_size*, on choisi les tranches à segmenter. *Volume* a donc une taille $M \times N \times P$.

```
Volume(:, :, :)=V1(:, :, Slice_start :Slice_start+Block_size-1);
```

On définit les variables *nb_iter* (nombre d'itérations dans l'IMC), *nb_phase* (le nombre de phase de la segmentation) et *mu* (le paramètre de régularisation).

```
nb_iter=20;nb_phase=7;mu=0.003;
```

On initialise les phases. '*middle*' signifie qu'on choisit la méthode choisissant les milieux des intervalles. Dans *C_init* (matrice ligne de *nb_phase* colonne) sont stockés les valeurs moyennes des phases et *P_init* ($M \times N \times P$) est la carte d'initialisation des labels

```
[C_init P_init]=initialisation_phases(Volume,nb_phase,'middle');
```

On segmente *Volume* par l'ICM avec la fonction **ICM3Dmulticanal**. En sortie, on a le volume segmenté *Sumphase* ($M \times N \times P$), la carte des labels correspondante *LabelICM* ($M \times N \times P$), les valeurs moyennes finales des phases *Cf* ($1 \times nb_phase$) et les phases séparées dans *Phase* ($M \times N \times P \times nb_phase$)

```
[Sumphase Phase LabelICM Cf]=ICM3Dmulticanal(Volume,nb_phase,mu, nb_iter, C_init);
```

Segmentation multicanal

Pour une segmentation multicanal, on utilisera les mêmes fonctions sauf que certaines variables n'auront pas les mêmes dimensions. Voici un exemple.

Lecture des fichiers *nom du fichier1* pour obtenir les volumes *V1* et *V2*.

```
[V1 HeaderI]=read_volume_img(nom du fichier1,'b');  
[V2 HeaderI]=read_volume_img(nom du fichier2,'b');
```

Avec les variables *Slice_start* et *Block_size*, on choisi les tranches à segmenter. *Volume* a donc une taille $M \times N \times P \times 2$.

```
Volume(:, :, :,1)=V1(:, :, Slice_start :Slice_start+Block_size-1);  
Volume(:, :, :,2)=V2(:, :, Slice_start :Slice_start+Block_size-1);
```

On définit les variables *nb_iter* (nombre d'itérations dans l'IMC), *nb_phase* (le

nombre de phase de la segmentation) et μ (le paramètre de régularisation).
`nb_iter=20 ;nb_phase=7 ;mu=0.003 ;`

On initialise les phases. '*middle*' signifie qu'on choisit la méthode choisissant les milieux des intervalles. Dans *C_init* (matrice de taille $2 \times \text{nb_phase}$) sont stockés les valeurs moyennes des phases et *P_init* ($M \times N \times P$) est la carte d'initialisation des labels

`[C_init P_init]=initialisation_phases(Volume,nb_phase,'middle');`

On segmente *Volume* par l'ICM avec la fonction **ICM3Dmulticanal**. En sortie, on a le volume segmenté *Sumphase* ($M \times N \times P \times 2$), la carte des labels correspondante *LabelICM* ($M \times N \times P \times 2$), les valeurs moyennes finales des phases *Cf* ($2 \times \text{nb_phase}$) et les phases séparées dans *Phase* ($M \times N \times P \times \text{nb_phase}$)

`[Sumphase Phase LabelICM Cf]=ICM3Dmulticanal(Volume,nb_phase,mu,nb_iter, C_init);`

Liste des fonctions

Fonctions pour l'initialisation des phases

`initialisation_phases` : fonction principale pour initialiser les phases (appelle les fonctions du dessous)

`initialisation_inf` : initialiser un canal avec les bornes inférieures

`initialisation_middle` : initialiser un canal avec les milieux.

`initialisation_sup` : initialiser un canal avec les bornes supérieures.

`initialisation_phases_multicanal` : initialiser les phases en multicanal

`carte_initialisation` : calcule la carte d'init à partir des valeurs moyennes des phases.

Fonctions pour l'ICM

`ICM3Dmulticanal` : fonction principale pour l'ICM

`regularisation` : calcule la régularisation dans la fonction ICM3Dmulticanal

Fonctions pour le recuit simulé

`Metropolis` : algorithme du recuit simulé

`regularisation_recuit` : calcule la régularisation dans la fonction Metropolis.

Fonctions pour études

`calcul_indices` : calcule les vrai et faux positifs, les indices de Jaccard et de similarité

`calc_volume_label` : calcule le volume d'une phase.

`energy` : calcule l'énergie de l'image segmentée.

```

function[Sumphase Phase Label Cf]=ICM3Dmulticanal(Volume,nb_phase,mu,nb_iter,C_init)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
**** fonction permettant de segmenter un volume d'images.
****
**** Volume (dimensions M*N*P*Q) : Volume a segmenter. M*N dim en pixel d'une coupe, P le
****                               nombre de coupes et Q le nombre de canal
**** nb_phase : le nombre de phase utilisé pour segmenter.
**** mu       : valeur du parametre de regularisation
**** nb_iter  : nombre d'iteration maximale dans l'algorithme
**** C_init   : valeur moyenne des phases
****
**** Sumphase : image segmentee (dim:M*N*P*Q) avec les valeurs moyennes des phases
**** Phase    : (dim:M*N*P*nb_phase) images de chaque phase
**** Cf       : valeurs moyennes des phase (dim:Q*nb_phase) dans chaque canal
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[M N P Q]=size(Volume);

%Initialisation de tous les tableaux utilises dans le programme
Label=zeros(M,N,P);
Sumphase=zeros(M,N,P,Q);
Phase=zeros(M,N,P,nb_phase);
dist=zeros(M,N,P,nb_phase);
count=zeros(nb_iter,nb_phase);
Number_label=zeros(1,nb_phase);
longdist=zeros(nb_phase,M*N);
d=zeros(1,Q);
nor=zeros(1,Q);
Mean=ones(Q,nb_phase);
Number_label=ones(1,nb_phase);
iter=0;
iter_stop=0;
reinit=0;

C(1,:,1)=C_init(1,:);
if Q==2
    C(1,:,2)=C_init(2,:);
end
for q=1:Q
nor(q)=(max(max(max(max(Volume(:,:,:,q)))))-min(min(min(min(Volume(:,:,:,q))))))^2;
end
if Q==2
nor(2)=nor(2)*1;
end

% Debut de la boucle principale sur le nombre d'iteration de l'algorithme
while iter_stop==0 && iter<=nb_iter
    iter=iter+1

    % Calcul de l'energie d'heterogeneite
    for i=1:M
        for j=1:N
            for p=1:P
                for k=1:nb_phase
                    for q=1:Q
                        d(q)=(Volume(i,j,p,q)-C(iter,k,q))*(Volume(i,j,p,q)-C(iter,k,q))/nor(q);
                    end
                    if Q==2
                        dist(i,j,p,k)=max(d(1),d(2));
                    else
                        dist(i,j,p,k)=d(1);
                    end
                end
            end
        end

        % Calcul de l'energie de regularisation
        L=regularisation(Label,8,mu,nb_phase);

        % Addition des deux energies
        dist=dist(:,:,:,1:nb_phase)+L(:,:,:,1:nb_phase);
    end
end

```

```

% Conversion du tableau dist (4D) en un tableau longdist (2D) pour
%accroitre la vitesse de calcul
for p=1:P
    for j=1:N
        for i=1:M
            for k=1:nb_phase
                longdist((p-1)*nb_phase+k,(j-1)*M+i)=dist(i,j,p,k);
            end
        end
    end
end

% Choisit la phase qui minimise l'energie au pixel considere
for p=1:P
    [F,I]=min(longdist((p-1)*nb_phase+1:p*nb_phase,1:end));
    Etot(iter,p)=sum(F);
    for j=1:N
        Label(1:M,j,p)=I((j-1)*M+1:j*M);
    end
end

% Conditions de reinitialisation de la valeur moyennes des phases
if iter>1
    if abs(Etot(iter)-Etot(iter-1))<=0.3
        reinit=1;
    end
end
reinit=1;

% Calcul des nouvelles moyennes des phases
if reinit==1
    for k=1:nb_phase
        BIN=double(Label==k);
        Number_label(iter,k)=sum(sum(sum(BIN)));

        for q=1:Q
            Mean(q,k)=sum(sum(sum(Volume(:,:,q).*BIN)));
            Mean(q,k)=ceil(Mean(q,k)./Number_label(iter,k));
            C(iter+1,k,q)=Mean(q,k);
        end
    end
    reinit=0;
else
    for k=1:nb_phase
        for q=1:Q
            C(iter+1,k,q)=C(iter,k,q);
        end
    end
end

% Calcul des images segmentees
for p=1:P
    for i=1:M
        for j=1:N
            Sumphase(i,j,p,1)=C(iter+1,Label(i,j,p),1);
            if Q==2
                Sumphase(i,j,p,2)=C(iter+1,Label(i,j,p),2);
            end
        end
    end
end

% Fin de la boucle principale
Cf(1,:)=C(iter+1,:,1);
if Q==2
    Cf(2,:)=C(iter+1,:,2);
end

% Calcul des phases
for p=1:P
    for k=1:nb_phase
        Phase(:, :, p,k)=Cf(1,k)*double(Sumphase(:, :, p,1)==Cf(1,k));
    end
end
end

```

```

function L=regularisation(Label,connexite,mu,nb_phase)

[M N P]=size(Label);
L=zeros(M,N,P,nb_phase);

switch connexite
case 4
    if P==1
        Mask=[0 1 0; 1 0 1; 0 1 0];
    else
        Mask(:,:,1)=[0 0 0;0 1 0;0 0 0];
        Mask(:,:,2)=[0 1 0;1 0 1;0 1 0];
        Mask(:,:,3)=[0 0 0;0 1 0;0 0 0];
    end
case 8
    a=1/sqrt(2);
    if P==1
        Mask=[a 1 a; 1 0 1; a 1 a];
    else
        b=1/sqrt(3);
        Mask(:,:,1)=[b a b;a 1 a;b a b];
        Mask(:,:,2)=[a 1 a;1 0 1;a 1 a];
        Mask(:,:,3)=[b a b;a 1 a;b a b];
    end
end

norma=sum(sum(sum(Mask)));

for k=1:nb_phase
    Label2=double((Label~=k));
    L(:,:,k)=(mu/norma)*convn(Label2,Mask,'same');
end

```

H Interface graphique pour segmenter des images par l'ICM

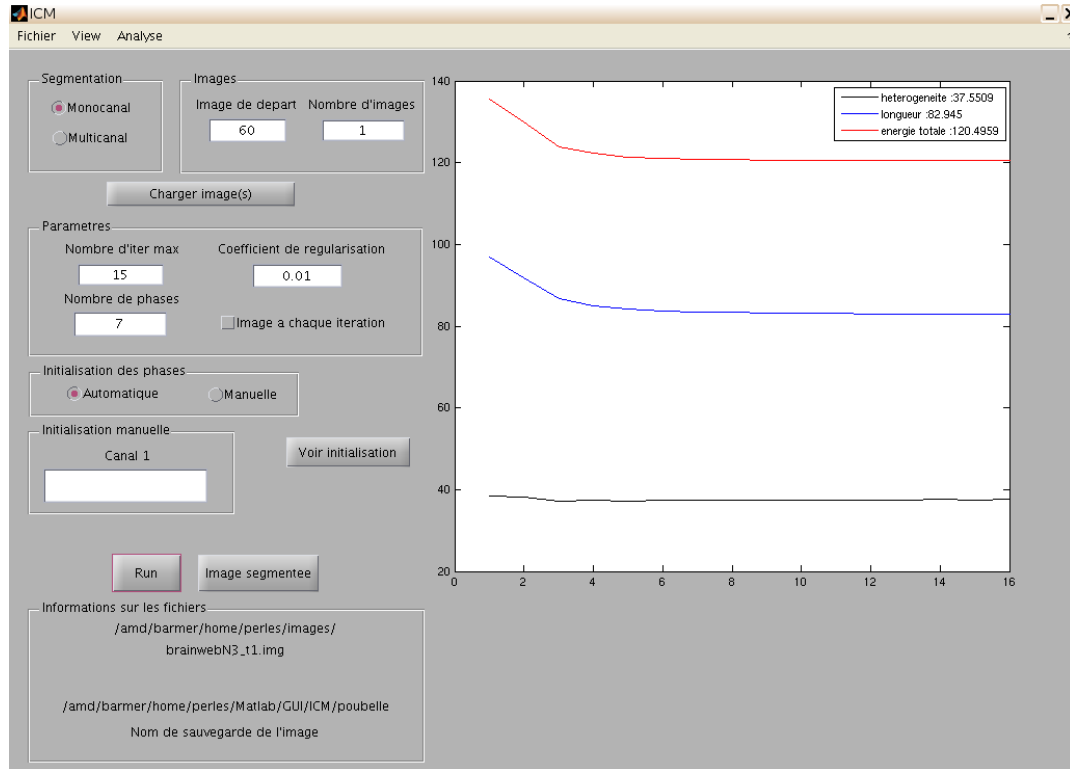


FIG. 42 – Cette interface permet d'utiliser simplement l'algorithme. On peut choisir l'image à segmenter, le répertoire de destination pour sauvegarder les résultats. L'interface permet de définir tous les paramètres et notamment de visualiser l'initialisation des phases avant de lancer l'algorithme. Sur le graphique de la fenêtre principale s'affiche l'évolution de l'énergie au cours des calculs. Enfin on peut visualiser les images segmentées et les phases obtenues dans le menu *View*