Image segmentation using socials agents

Richard Moussa, Marie M. Beurton-Aimar, Guilhelm Savin, Pascal Desbarats

Laboratoire Bordelais de Recherche en Informatique
UMR 5800 - Université Bordeaux I, 351, cours de la Libération,
33405 Talence CEDEX, France

Research Report

Image segmentation using social agents

Richard MOUSSA, Marie BEURTON-AIMAR, Guilhelm SAVIN and Pascal DESBARATS

December, 2008

Image segmentation using social agents

Abstract

In the literature, there are many methods for image segmentation. Many methods are often limited to a few number of images or by their sensibility to image artefacts like noise. Thus, there is no generic method for solving problems of image segmentation. In this report, we will introduce a new way to improve segmentation methods which use a model coming from biology: the social spiders and implemented by a multi-agent system. This method developed will be presented with its shortcomings and the way to overcome them. Finally, some perspectives will be shown to improve the method and its results on MRI[1] images.

Keywords: Image segmentation, Social spiders, Multi-agent system, Articial life.

[1] Magnetic Resonance Imaging

Contents

Introduction

Image segmentation is one of the most difficult task in digital image processing. This is an important step for quantitative analysis of brain images and to study many brain disorders. Indeed, structural changes in the brain can be the result of brain disorders. The quantification of these changes, by measuring area of regions of interest, can be used to characterize disease severity or evolution. Manual marking out of cerebral structures in MRI[2] images by an expert is obviously a time consuming process. Moreover, these manual segmentations are prone to large intra/inter-observer variability. Thus, the segmentation remains a challenging task. Image segmentation algorithms subdivide images into their constituent regions, with the level of subdivision depending on the problem to be solved. Robust, automatic image segmentation requires the incorporation and efficient utilization of global contextual knowledge. However, the variability of the background, the versatile properties of the target partitions that characterize themselves and the presence of noise make it difficult to accomplish this task. Considering the complexity, we often apply different methods in the segmentation process according to the nature of the images.

Social spiders belong to spider species whose individuals form relatively long-lasting aggregations. Whereas most spiders are solitary and even aggressive toward conspecifics, hundreds of species show a tendency to live in groups and to develop collaborations between each other, often referred to as colonies. For example, these spiders of 5mm in body length are capable to fix silks up to a volume of $100m^3$ [6]. This technique is used to trap preys having big forms.

This work shows how it is possible to use the social spiders method to implement an efficiency image segmentation method to treat brain MRI images.

This report is organized as follows: the first section gives a view on image segmentation methods. The second section describes different types of multi-agent systems, their utilities for the image segmentation. The third section presents some related works with this subject. The following section presents the method used for segmenting an image before we provide an experimental assessment of the method. Finally, we will conclude and propose some ideas for further works.

1 Image Segmentation

Image segmentation consists on partitionning an image into a set of regions that covers it. After this process, each pixel/voxel is affected to a region and each region corresponds to a part of the image. The discontinuity between the regions constructs the contour of the object. The segmentation approaches can be divided into three major classes [7]. The first corresponds to pixel-based methods which only use the gray values of the individual pixels. The second is region-based methods which analyze the gray values in larger areas for detecting regions having homogenous caracteristics, criteria or similitude.

[2]Magnetic Resonance Imaging

Finally, the edge-based methods detect edges and then try to follow them, this can be done by computing a luminacy function. The common limitation of all these approaches is that they are based only on local information. Even they can only use a part of the information. Pixel-based techniques do not consider the local neighborhood. Edge-based techniques look only for discontinuities, while region-based techniques only analyze homogeneous regions.

2 Multi-agent system

As we have previously explained, a multi-agent system is a distributed system composed of a group of agents, which interact between them through an environnement. Agents are classified in two categories: cognitive and reactive. Cognitive agents have a global view of the environment, they know the task for which they work. Conversely, reactive agents only know a restricted environment part. They react to environment stimulus and can modify this environment by adding or removing informations. Reactive agents do not know the complex task for which they work: they have a restricted set of simple features and they only apply them. Insects colonies like spiders and ants are an example of reactive agents: each one knows locally what it has to do, but no one knows the more complex task for which they work. Such insects are called social insects.

Domain: Artificial Life Vs Artificial Intelligence

The most part of multi-agent systems are considererd as relevant to Artificial Life Domain. To clarify the field of Artificial Life (AL), we will distinguish it from the Artificial Intelligence (AI): AI takes human cognition as a reference and is interested in designing machines that can simulate human cognition. AL tends to be closer to biological functions and processes over behavior "reflexes" than logical reasoning or cognitive acts. Its field of study is broader, it explores the characteristics of living in general. AL operates the concepts from the current cognitivity and connectivity that the cognitive science share. Therefore, it proposes solutions based sometimes on the symbolic approach, sometimes on the self-organizational approach, or built its models from a combination of both methodologies. One can distinguish two types of work from the AL [1]. On the one hand, simulations using only the computer. It is recognized, in this case, that any system can be modeled by computer. That is to say a formal system is adequate to represent a physical system in a likely way. On the other hand, achievements, where the concrete and hardware system is paramount. It is considered here that the physical dimension of a system is irreducible to a symbolic representation.

3 Related Works

Ramos and Almeida [10] have explored the idea of using a digital image as an environnement for artificial ant colonies. They observed that artificial ant

colonies could react and adapt appropriately their behavior to any type of digital habitat. Ramos et al. [11] also investigated ant colonies based data clustering and developped an ant colony clustering algorithm which he applied to a digital image retrieval problem. By doing so, they were able to perform retrieval and classification succesfully on images of marble samples. Liu and Tang [8] have conducted similar works and have presented an algorithm for greyscale image segmentation using behavior-based agents that self reproduce in areas of interest. He and Chen [5] have provided an artificial cell model. In their model, each life is one individual unit, called a cell, which adheres to one pixel in the image. All the cells bear similar structures but mutations may occur during the process of reproduction due to the influence of the environment. Hamarneh et al. [4] have shown how an intelligent corpus callosum agent, which takes the form of a worm, can deal with noise, incomplete edges, enormus anatomical variation, and occlusion in order to segment and label the corpus callosum in 2D mid-sagittal images slices in the brain. Bourjot et al. [2] have explored the idea of using social spiders as a behavior to detect the regions of the image. The principle is to weave a web over the image by fixing silks between pixels. This method has been selected, implemented (see section 4) and experimented (see section 5) in comparaison with other type of methods.

4 Segmentation Method

The multi-agent system is composed of an environment and a set of agents. For segmentation purpose, environment is created from a given grayscale picture: it is a matrix of gray pixels.
System and agents have a life cycle. Figure 1 shows that a cycle of the system consists in executing the life cycle of each agent. This life cycle is transposed to a step. The number of steps to be executed is given by the user. Algorithm 1 presents a description of figure 1 where for a given number of iterations (steps), each spider computes the three functions. Its complexity is about $O(Ite.N)$, where Ite is the number of iterations and N the number of spiders.

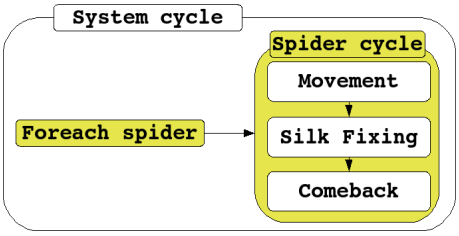

Figure 1: System overview

Algorithm 1 Segmentation method

Require: Pixels: Matrix of pixels $\in \mathbb{N}^2$, Width and Height $\in \mathbb{N}$, Ite $\in \mathbb{N}$ and Conf: Configuration parameters

```
scuts[Width][Height]: Draglines matrix
Create colonies and spiders from the Conf
while Ite - - > 0 do
   for Each spider s do
      Movement(s, Pixels, scuts)
      Skillfixing(s, scuts)
      Comeback(s)
   end for
end while
```

Environnement

Environment is a matrix of gray pixels. Each pixel is a position for spiders and allows them to access to other pixels. Neighborhood of a pixel p is defined in two ways (figure 2):

- The 8 pixels around p, which are named *Neigh*.
- All pixel linked to p by a dragline, which are named *SCuts*.

Set of all reachable pixels are named *Access*.

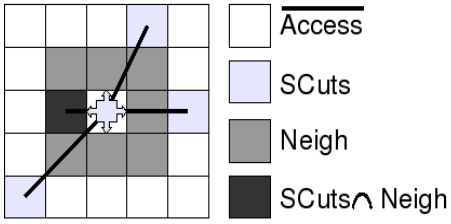

Figure 2: Neighborhood of a pixel

Agent

As previously mentionned, spiders are reactive agents. They are defined by an internal state composed of a set of parameters values, a current position and the last pixel where a spider has silked. These spiders have also an ability to move in the environnement, to fix silk[3] and to come back[4].

Spiders which detect the same region can be grouped in a set called a colony. Spiders of a same colony share the same set of parameters values.

An agent life cycle, as shown is figure 1, consists in firing each behavioral item according to a probability which depends of parameters values and environmental characteristics at the spider position.

[3] Weave a dragline between two pixels
[4] Return to the last fixed pixel

Movement

Movement is computed in two ways: the way where only one colony of spiders is set up in the environment (single region detection) and the way where several colonies try to detect a region (multiple-region detection). In both cases, a spider reaches one neighbor of its current pixel like described in figure 2.

In the first case, spider must choose between move to a pixel in *SCuts* or move to a pixel in *Neigh*. This choice is done according to the value of simulation parameter *pdraglines*. Then, each pixel of the choosen set has the following probability to be choosen:

- if set is *Neigh*, each pixel has the same probability
- if set is *SCuts*, each pixel p has a probability $\frac{draglines_p}{draglines_{all}}$ where $draglines_p$ is the number of draglines going to p and $draglines_{all}$ is the number of draglines starting from the current position of the agent.

Probability of a pixel which is in both set is the sum of both probabilities.

In the second way, a weight-function is used to compute the probability to move to a pixel. So, for each pixel p, probability to move to p is

$$P(Move(p)) \;=\; \frac{w(p)}{\sum_{a \in Access} w(a)} \tag{1}$$

$w(p)$ is constant if p is in *Neigh*, else if p is in *SCuts* we distinguish draglines woven by agents colony and those woven by other colonies. Two simulation parameters, *attractself* and *attractother*, are used to compute this probability. These parameters are respectively the attraction for draglines woven by agents colony and draglines woven by other colonies. A function F is also used to express draglines-counts influence on weight. So for a pixel p in *SCuts*, we have the following weight :

$$\begin{aligned} self &= attractself \;.\; F(draglines_{self}) \\ other &= attractother \;.\; F(draglines_{other}) \\ w(p) &= self \;+\; other \end{aligned} \tag{2}$$

Algorithm 2 describes the movement function by affecting to the weights of the neighborhood pixels whether the weight of the colony of the spider whether its weight function and finally move the spider to a pixel from the neighborhood according to its weight value. Its complexity is $O(Nei)$ where Nei is the maximun of neighborhoods of a pixel.

Algorithm 2 Movement

Require: S: Spider, Pixels: Matrix of pixels $\in \mathbb{N}^2$, Scuts: Matrix of draglines $\in \mathbb{N}^2$

1: weights[neightSize(position(S)) + scutsSize(position(S))]: weights of the neighborhood pixels
2: **for** i $\in$ {0, ..., neightSize(position(s)) - 1} **do**
3: weights[i] $\leftarrow$ w(p)(colony(S))
4: **end for**
5: **for** i $\in$ {neightSize(position(S)), ..., Size(weights) - 1} **do**
6: weights[i] $\leftarrow$ WeightFunction(S)
7: **end for**
8: Choose a pixel in the neighborhood according to its weight and move the spider to the pixel choosen

Silkfixing

Here, two other simulations parameters are used, *reflevel* and *selectivity*. The first one corresponds to the graylevel of the region to detect, and the second one defines the tolerance to fix a dragline with a pixel whose graylevel is not exactly reflevel.
Probability to fix a dragline with current pixel (figure 3) follows a gaussian distribution whose mean is *reflevel* and standard deviation is $\frac{1}{selectivity}$. Algorithm 3 consists on choosing a random number and evaluating it with a gaussian function to add or not a dragline. Its complexity is constant.

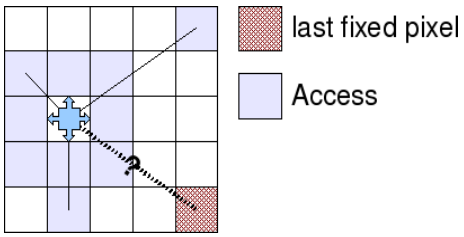

Figure 3: Fix a dragline with current pixel

Algorithm 3 Silkfixing

Require: S: Spider, Pixels: Matrix of pixels $\in \mathbb{N}^2$, Scuts: Matrix of draglines $\in \mathbb{N}^2$

1: p $\leftarrow$ random(0, 1)
2: **if** p $<$ Gauss(level(position(S))) **then**
3: Add a dragline between postion(S) and lastfixed(S)
4: **end if**

Comeback

This behavorial item allows spiders to come back to the last silked pixel. This action is fired depending on the probability value defined as *backprobability*.

The aim of this action is to detect connected region of pixels: spiders can not go far of pixels they have silked. As it is shown by [2], disable this item leads to create draglines between two unconnected groups of pixels. Algorithm 4 compares a random number with a probability function and tries to return to last fixed dragline if the condition is satisfied in ligne 2. Its probability is also constant.

Algorithm 4 Comeback

Require: S : Spider
1: p ← random(0, 1)
2: **if** p < backprobability(colony(S)) **then**
3: postion(S) ← lastfixed(S)
4: **end if**

4.1 Dynamic system

Dynamic system is based on stigmergy: each spider lets informations on the environment, those are used by other agents or by itself in a next cycle. Image segmentation emerges from the global task achieved by all the spiders: after a certain number of system iterations, draglines are created. Degree of a pixel defines number of draglines coming in or out of this pixel. In multiple-regions detection, degree is given according to a colony. In this case, global degree is the sum of degree for each colony. Region detected by a colony is composed of pixels having higher degree for this colony.

4.2 Problems

The social spiders method raises two important problems :

1. There is a high number of parameters
2. Computing the number of iterations required or defining a stop condition

The parameters computation can be problematic if it is done empirically. Indeed, to compare the results of social's spiders segmentation among several images, we must be sure that the computation of the parameters will be in an equivalent manner in all cases.

Similarly, it is important that the stop condition meets the same criteria between different acts of segmentation. Otherwise, the results could be evaluated: a number of iterations not big enough lead to a poor qualitative analysis, and a number of iterations too high could increase the simulation time of the method without improving the quality of the results.

4.3 Solutions

In this section, we propose solutions to the two problems evaluated above.

4.3.1 Autodetection of parameters

The method of socials spiders has several parameters that will define the regions to be detected. These parameters are presented in Table 1.

Parameter	Description
reflevel	greyscale of the colony
selectivity	acts on the probability to weave a dragline from the current position of the insect
attractself	attracts toward the dralines of the same colony
attractother	attracts toward the dralines of the other colony
backprobability	probability to turn back
$w(p)$	weight of a pixel with a direct neighborhood
saturationvalue	maximun weight of a pixel with an indirect neighborhood

Table 1: Colony parameters.

First, it must know the number of colonies (the number of type of regions) and each of the properties, in particularly the grayscale reference and the *selectivity* parameter. We have seen that the computation of these empirical parameters could be an obstacle for comparing the results producted by the socials spiders method: if the parameters are not computed in the same way in each case, the comparisons became unreliable because they depends on the computation method. We'll see in this part a method for determining the parameters of the simulation. This method will be used thereafter in all segmentations processes.

Each colony has a grayscale (or intensity) of reference which will serve the spiders of the colony to determine if they should fix a silk. There is two possibilities:

1. Determining "manually" the intensities, which is unattractive except where the intensities are known by the user;

2. Using a method to determine automatically the intensities.

The automatic detection will allow us to obtain optimal parameters without the user knowledge about the image. Thereafter, we will appoint an intensity level of pixels/voxels of this intensity.

We will use the histogram of the image to determine the intensity of interest to be detected. Indeed, a region in a image usually implies a peak more or less important in the histogram of the image. Therefore, we will detect successively each peak by eliminating the peak intensities component of future options. The Figure 4 shows the detection of maximum (peak heights) of the histogram:

A: detecting a maximum intensity level n from the histogram.

B: detecting the neighborhood intensities v such as:

- If v < n then v + 1 is detected and degree (v) $\leq$ degree (v + 1)
- If v > n then v - 1 is detected and degree (v) $\leq$ degree (v - 1)

C: marking the intensities detected in order to not use them later

D: returning to step A until all the intensities have been marked

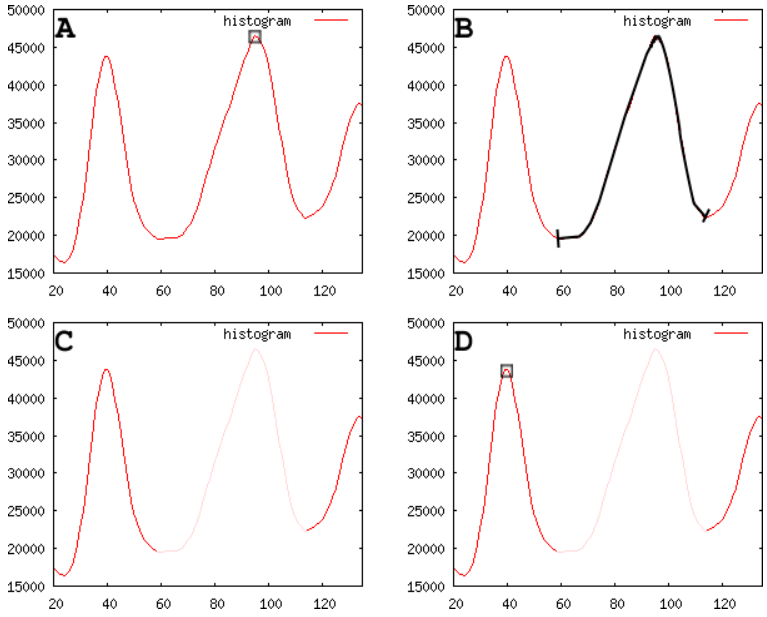

Figure 4: Detection of the maximas in the histogram.

This method allows us to obtain a list of maximas in the histogram.

However, this method is particularly sensitive to noise at Step B. Therefore, we will introduce a method to reduce the noise effects on the histogram by smoothing it and allowing to obtain a maximum number of representative of the number of classes intensities of the image.

Histogram smoothing

Histogram smoothing will reduce the peaks caused by noise (see figure 5). The method proposed here is done by iteration. At each iteration, the degree of intensity n becomes the average degrees of intensity n - 1, n, n + 1. We propose a method that detect automatically the optimal settings for the segmentation of the image. The method determines the number of settlements and the parameters *reflevel*, *selectivity* and the number of spiders in each colony.

First, We will determine the maximas in the histogram as described above. Then, the histogrtam will be smoothed until fewer maximas are significant.

Indeed, the maximas caused by noise are on the peaks of the histogram whose slopes are small and will be erased by smoothing them in few iterations. Finally, the maxima representing a region is located on a peak having an important slope which requires a large number of smoothing in order to be eliminated. However, the risk to clear up an interesting maxima is not zero.

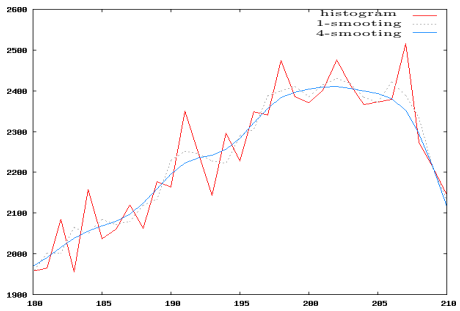

Figure 5: Histogram smoothing

In order no to fall into aberrations, we add the condition that the number of maximas, and therefore the number of regions to detect, must be greater than 2. Indeed, the image contains at least a substance and a foreground.

We get a serie of maximas M_1, M_2, ..., M_k. The number of maximas, k, determines the number of spiders colonies that will be used. Each parameter is the maximum *reflevel* of the colony. Then, we will partition the intensities in as many classes as maximas detected. To do this, simply find the minimum level of intensity between two maximas. This yields a serie of minimas m_1, m_2, ..., M_{k-1} so that $0 < M_1 < m_1 < \ldots < m_{k-1} < M_k$ ¡ l where l is the maximum intensity. The k classes intensities obtained are $[0; m_1]$,$]m_1; m_2]$, ..., $]m_{k-1}; l]$.

The variance of each class provides the parameter *selectivity* of each column. The number of spiders per colony may be adjusted according to the sum of degrees of class.

We can now determine the essentials parmaters of the socials spiders method. It remains to determine the numbers of spiders per region and the attraction of the silks on spiders.

4.3.2 Stop condition

We have seen that the socials spiders method has a life cycle which is repeated a number of times until the image is segmented. The number of iterations will influence two important points on the result of segmentation:

1. The quality of segmentation;

2. The execution time required to achieve this result.

At each iteration, the spiders will weave between the pixels that will be used to determine to which region is the pixel. If the number of iteration is not

enough, the number of pixels that do not belong to any region will be important and the result will be of poor quality. On the contary, if the number of iterations is too large, the spiders will only increase after a certain time the silks already existing without providing any new information. This last point will have as effect a longer execution time for an identical quality result.

As for the parameters, the computation of the number of iterations is an important point to obtain comparative results that are credible. Rather than fixing a number of iterations, it is possible to determine a stop condition to be verified before an iteration.

Definition Let β be the number of silks fixed between pixels whose degree is zero during an iteration.

The result of socials spiders segmentation depends on the silk which will be fixed between the pixels. During an iteration, when β tends towards zero, we can consider that the system stabilizes and the spiders only reinforce the existing silks.

Figure 6 shows the evolution of β along the segmentation process.

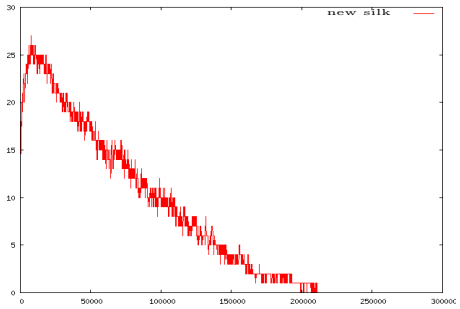

Figure 6: Number of silks fixed during an iteration

It is possible to detect when β remains at zero. This moment determines to stop the simulation. We can improve this condition by adding two parameters:

- A threshold that determines when β may be considered invalid;
- The number of authorized β zero before stopping the simulation.

The threshold may be determined by the number of spiders. Indeed, at each iteration, each spider has the possibility to fix a silk. The number of silks fixed during an iteration is bounded by the number of spiders.

5 Results

In this section, we will present the results obtained by our simulation on MRI images. First, we will present the image used to improve our results. Then, we will describe the methods used for comparaison. Finally, we will show the results of the simulations and argue them.

5.1 BrainWeb

BrainWeb[5] is a simulator of MRI brains images. It helps to validate technical medical analysis as segmentation. Indeed, the image is obtained from a model which we know its different components. A slice of this model is presented figure 7.

Figure 7: Slice of the 3D model

We can distinguish ten items in this model are listed in Table 2.

Element	Pourcentage	Form
background	42.2%	
csf	5.2%	
grey matter	12.7%	
white matter	9.5%	
fat	2.1%	

Element	Pourcentage	Form
muscle skin	8.7%	
skin	10.2%	
skull	5.1%	
glial matter	0.1%	
connective	4.2%	

Table 2: BrainWeb Model: the various components.

The parameters of the model to be used in this study are:

- Modality: T1
- Slice thickness: 1mm
- Noise: 3%
- Intensity non-uniformity: 20%
- Dimensions: 181 x 217 x 181vx

First, we will segment a 2d slice of this image (Figure 8) and then attempt the same process on a 3D image. This slice choosen to our tests is the 90th slice of the image. This image is used to determine if the social spiders method is able to segment images having complex forms. The result obtained will allow us to decide if it is feasible to segment images in three dimensions.

Figure 8: 2D slice: 181 X 217 px

[5] http://www.bic.mni.mcgill.ca/brainweb/

5.2 Comparison methods

In this section, we will compare the social spiders method with other segmentation methods. These comparisons will allow us to determine whether the social spiders method brings something positive compared to traditional segmentation methods.

We will use these comparisons on two other methods:

- A classification method by thresholding: the otsu method;
- A region-based method: the region growing method.

To compare these methods, we need to establish criteria to be used on all test images. We will compare the results on several points:

1. The number of regions;
2. Correspondence between the regions of the model and the segmentation result;
3. Execution time.

Definition Let Γ be an image and Δ its segmentation result. We call Γ_i the region i of the image and Δ_j the region j of the result.

The number of regions will allow us to determine whether the method considered detects a number of regions close to reality. In the case of noisy images, it is possible that some methods detect regions with insignificant size. That is why we will add to the total number of regions, the number of regions having insignificant size.

Definition A region will be considered insignificant if its size is less than 10 pixels.

The computation of the number of regions will be done on the segmentation method result on which a labeling is added to the connected components to consider the regions connected.

The correspondence between initial image and its result will enable us to determine if the regions identified by the method correspond to the regions defined in the initial image. This is only possible in the case of images for which we know lots of informations. This match will be appointed accuracy.

To compute the accuracy, it is necessary to determine which region Δ_j matches the most the region Γ_i. This region is determined by:

$$\begin{aligned} n^i &= \frac{the\ total\ number\ of\ pixels}{the\ voxels\ of\ \Gamma_i} \\ n_j &= \frac{the\ total\ number\ of\ pixels}{the\ voxels\ of\ \Delta_j} \\ n_j^i &= \frac{the\ total\ number\ of\ pixels}{the\ common\ voxels\ between\ \Delta_j\ and\ \Gamma_i} \end{aligned} \tag{3}$$

Thus, it is possible to compute $\delta_{i,j} = \frac{n_j^i}{n^i}$ and $\gamma_{i,j} = \frac{n_j^i}{n_j}$ representing respectivly the proportion of pixels / voxels of Γ_i belonging to Δ_j and the proportion of pixels / voxels of Δ_j belonging to Γ_i. We have two ways to choose the region that corresponds to Δ_j corresponding the most to Γ_i:

1. Δ_k as the value of $\delta_{i,k}$ is maximum: in this case, we prefer the fact that Γ_i and Δ_j have a maximum of pixels / voxels in common;

2. Δ_k as $\delta_{i,k} + \gamma i, k$ is maximum: same as above, but we add the requirement that Δ_k must have a minimum of pixels / voxels in other regions than Γ_i.

In our results, we indicate two points, $accuracy_\delta$ and $accuracy_{\delta+\gamma}$, which corresponds respectively to the two choices of Δ_k described above. In both cases, the accuracy will be the average values for all regions of the model.

5.2.1 Region growing

The *Region growing* method as described by Shapiro and Stockman [12] consist on building a region from one chosen pixel and then adding recursively neighbors whose grayscale difference with the original pixel is below to a threshold.

This method tries to grow an initial region by adding to this region pixels that do not belong to any region but to the pixels neighboorhood already in the region and whose grayscale is sufficiently close to the area. When it is not possible to add pixels, we create a new region with a pixel that has not been selected yet, then we grow the region.

The method ends when all the pixels were chosen by a region. To automatize the process of this algorithm, the threshold was choosen as the local minima of the image histogram.

5.2.2 Otsu

Otsu [9] has developped a multi-level thresholding method. Its aim is to determine, for a given number of regions, the optimum values of different thresholds based on the variance of subdivisions created.

The basic method consists on separating the foreground from the backgroud. In this case, we search the optimal threshold to split the pixels in two regions. For a threshold t, it is possible to compute *the between-class variance* σ^2(t). This measure is derived from the average intensity μ_1, μ_2 and μ of classes [0; t], [t + 1; L] and [0; L] where L is the maximum intensity.

The Equation 4 shows us the computation of σ^2, where w_1 and w_2 represent the proportion of pixels in the class [0; t] and [t + 1; L] compared to the total number of pixels.

$$\sigma^2(t) = w_1(t)(\mu_1(t) - \mu)^2 + w_2(t)(\mu_2(t) - \mu)^2 \tag{4}$$

The Otsu method shows that the optimal threshold t^* is obtained for a between-class variance. The method consists on computing the variance for all possible thresholds (t $\in \{1; \ldots; L - 1\}$) and determining its maximal value.

This method could be extended easily to the computation of M classes with M - 1 thresholds $\{t_1; t_2; \ldots; T_{M-1} - 1\}$ $(t_1 < t_2 < \ldots < t_{M-1})$. The between-class variance is defined then as follows:

$$\sigma^2(t_1, \ldots, t_{M-1}) = \sum_{M-1}^{M} w_k(\mu_k - \mu)^2 \tag{5}$$

where w_k represent the proportion of pixels in the class $[t_{k-1}; t_k]$[6], μ_k the intensity average of this same classe and μ the intensity average of the class [0; L].

For each M-1-uplet, we compute thresholds of the between-class variance. The optimal thresholds, $(t^*, \ldots, t^*_{M-1})$, correspond to the maximum value of the between-class variance.

Chen et al. [3] propose an algorithm that minimizes the number of necessary computation to obtain a faster algorithm. This method had been implemented for the evaluations tests.

5.3 Experimentations

This section will explore if the socials spiders method supports complex images such as MRI images. For the 2D case, the results are presented in the table 3 and the figure 9.

The execution time to be given comes from the simulation of the methods on a machine equipped with two Intel (R) Xeon (R) E5345 (8 cores having 2.33GHz) and 8GB of RAM. The operating system of this machine is a Linux kernel 2.6.21 x86_64.

(a) (b) (c)

Figure 9: 2D segmentation of a brain slice: a) Social spiders, b) Region growing, c) Otsu

The Socials Spiders method produced the highest number of significant regions, that closest to the actual number of regions. However, it is also the

[6] $t_0 = 0$ and $t_M = \mathrm{L}$

	Region	Region > 10px	$Accuracy_{\sigma}$	$Accuracy_{\sigma+\gamma}$	Time
Social spiders method	4361	434	60.8%	54.9%	50 s
Region growing method	908	664	46.2%	21.1%	0.5 s
Otsu method	1875	567	90.7%	89.3%	0.2 s

Table 3: 2D results

method that produced the greatest number of insignificant regions, meaning that the spiders have oversegmented the image. This oversegmentation of the image has not been distented by the usage of a complex BrainWeb model. The execution time of the Socials Spiders method of spiders is too high compared to the two other methods. The number of silks fixed is less important because the image is composed of a greater number of regions, each region thus representing a smaller proportion of the image. Spiders make more time (number of iterations) to find pixels corresponding to their colony. The table 4 shows the results of different segmentations of the image BrainWeb in three dimensions.

	Region	Region > 10vx	$Accuracy_{\sigma}$	$Accuracy_{\sigma+\gamma}$	Time
Social spiders method	45575	13391	87.3%	87.3%	8 h
Region growing method	7866	1634	75.4%	46.7%	102 s
Otsu method	8497	865	83.9%	83.9%	10 s

Table 4: 3D results

The accuracy displayed by the spiders method is quite correct compared to the other methods. However, a number of regions rather high can be explained by a number of voxels non-detected more important, leading to disconnection of the regions.

Unfortunately, this accuracy has a very important consuming time over the execution time. Therefore, as well as for the results on two dimensions images, we get a bad time result. The Otsu method, although with a precision of a few percent lower, produces a good result for a shorter time computation of about 4000 times.

In addition, the number of regions produced by the spiders methods is greater by a factor of 10 compared to the other methods.

6 Discussion

In this report, we have presented a new method of segmentation that showed various problems of implementation. Indeed, this method requires a large number of parameters depending on the image to be processed. In addition, the method did not really have a stop condition which impose the user to fix in advance the number of iterations required.

First, we solved the problem of computation of the parameters by proposing a method to compute automatically these parameters. This method can determine the most important parameters of the method as follows:

- The number of colonies;
- The grayscale reference of each colony;
- The *selectivity* parameter of each colony.

The other parameters were set as they have less importance in the segmentation result.

Then, we also saw a method based on the number of silks fixed at an iteration to determine a stop condition to the method. This stop condition has allowed us to eliminate the problem of the number of iterations needed to produce a good segmentation.

We have made comparaisons between the results of the social spiders method, the region growing method and the otsu method. These comparisons focused on the accuracy of methods, the number of regions produced and the time processing of the methods. They are not exhaustive comparisons in the sense that all aspects of segmentation are not taken into account.

Through these comparisons, we have put forward some drawbacks on the socials spiders method. Particularly, we have seen that this method produced a significant number of areas and that the execution time was particularly long.

However, the socials spiders method is based on an architecture that is ideal to be parallel. This method is composed of a group of agents that can be spread over several processors. The fact that they all run the same algorithm suggests using a SIMD[7] architecture and therefore using this method on graphical processors (GPU[8]).

We have seen that the second drawback of the socials spiders method is producing an oversegmentation of the image. It is possible to solve this problem by initiating the algorithm by merging insignificant regions to ratach them to significant regions or merging them as appropriate. Merging operation is not an expensive treatment time processor, therefore, adding a post-treatment to the social spiders method adds a negligible computation time compared to the method itself.

Some solutions have been considered to improve the method. The spiders seem sensitive to the topology of the image, so it is possible to guide the movement of spiders with a gradient or a laplacian. Indeed, these measures will provide informations on the possible presence of contours. It would be then possible to use spiders in two ways:

1. Gradient would be repellent which would partition a colony of spiders in a region;
2. On the contrary, the gradient could have an attractive effect. In this case, spiders would be used to detect the contours of regions.

[7]Single Instruction Multiple Data
[8]Graphical Processing Unit

Conclusion

Finally, we saw a first approach to image segmentation in three dimensions using the socials spiders method. The method has been used without other modification than the extension of the neighborhood. Therefore, there is no optimization related to adding an extra dimension. However, the BrainWeb segmentation model showed that the method was capable to produce a result with good accuracy.

Optimizations are needed to reduce the number of regions produced and specially to reduce the time process of the method. In the previous section, solutions have been proposed to resolve or at least reduce these defects.

References

[1] M. Bedau. Artificial life: Organization, adaptation and complexity from the bottom up. *Trends in cognitive sciences*, 7(11):505–512, 2003.

[2] C. Bourjot, V. Chevrier, and V. Thomas. A new swarm mechanism based on social spiders colonies: from web weaving to region detection. *Web intelligence and Agent Systenms: An International Journal - WIAS*, 1(1): 47–64, March 2003.

[3] T.-S. Chen, P.-S. Liao, and P.-C. Chung. A fast algorithm for multilevel thresholding. *Journal of Information Science and Engineering*, 17:713–727, 2001.

[4] G. Hamarneh, T. McInerney, and D. Terzopoulos. Deformable organisms for automatic medical image analysis. *Medical Image Analysis*, pages 66–76, 2001.

[5] H. He and Y. Chen. Artificial life for image segmentation. *Internation Journal of Pattern Recognition and Artificial Intelligence*, volume 15, Issue 6:989–1003, 2001.

[6] E. Jackson. Social spiders. *Current Biology*, 17(16):R650 – R652, 2007.

[7] B. Jähne. *Digital Image Processing*. Springer, sixth edition, 2005.

[8] J. Liu and Y. Tang. Adapive image segmentation with distributed behavior-based agents. *IEEE Transactions Pattern Analysis and MAchine Intelligence*, volume 21, Issue 6:544–551, 1999.

[9] N. Otsu. A threshold selection method from gray level histograms. *IEEE Trans. Systems, Man and Cybernetics*, 9:62–66, 1979.

[10] V. Ramos and F. Almeida. Artificial ant colonies in digital image habitats - a mass behaviour effect study on pattern recognition. *Proceedings of ANTS2000 - 2nd International Workshop on Ant Algorithms (From Ant*

Colonies to Artificial Ants), in Marco Dorigo, Martin Middendorf and Thomas Stzle (Eds.), pages 113–116, Brussels, Belgium, 7-9 Septemnber 2000.

[11] V. Ramos, F. Muge, and P. Pina. Self-organized data and image retrieval as a consequence of inter-dynamic synergistic relationships in artificial ant colonies. *Frontiers in Artificial Intelligence and Applications, Soft Computing Systems - Design, Management and Applications, 2nd International Conference on Hybrid Intelligent Systems, IOS Press, in Javier Ruiz-del-Solar, Ajith Abraham and Mario Kppen (Eds.)*, volume 87, ISBN 1 5860 32976:500–509, Santiago, Chile, December 2002.

[12] L. Shapiro and G. Stockman. *Computer Vision*. Prentice-Hall, 2001.