# A multi-agent approach for the edge detection in image processings.

Jason MAHDJOUB[a]- jason.mahdjoub@leri.univ-reims.fr

Zahia GUESSOUM[ab]- zahia.guessoum@lip6.fr
Fabien MICHEL[a]- fmichel@leri.univ-reims.fr
Michel HERBIN[a]- michel.herbin@leri.univ-reims.fr

[a] *CReSTIC-MODECO, University of Reims, rue des Crayères, 51100, Reims, France*
[b] *LIP6, University of Paris 6, UPMC 4 place Jussieu, Case 169, 75252 Paris Cedex 05, France*

**Abstract**

Several multi-agent approaches have been proposed to improve image processing. They use several image processing algorithms simultaneously. However, these approaches do not deal with the inherent problems encountered for the extraction from an image of primitive information like edges or regions. This implies that agents use macro results provided by image processing algorithms. Agents use macro results provided by image processing algorithms. Then, the results do not take advantage of all the interesting characteristics, such as environmental adaptability and emergent behavior capability, of agent-based systems: the combinative explosion of the possible solutions offered by this kind of systems, is highly reduced.

In this paper, we propose a multi-agent system based on instinctual [5] reactive agents, which are able to detect edges. Agents locally perceive their environment, that is to say, pixels and additional environmental information. This environment is built using a Kirsch derivative and a Gradient Vector Flow. Edges detection emerges from agents interaction. Problems of partial or hidden contours are solved with the cooperation between the different agents. In the scope of this paper, we illustrate our approach through an example that shows how it can be used to detect lungs on 2D images coming from a scan device.

**Keywords :** multi-agent system, image processing, edge detection, reactive agents, Gradient Vector Flow

## 1 Introduction

The image processing domain and particularly the image segmentation aims to extract information from a picture[1]. At a first level and without considering frequential spaces or wavelets spaces, we can specialize this extraction as being the detection of two primitives [6]: (1) the edge (or contour) detection and (2) the region detection. A third category consists in detecting both regions and edges at the same time.

A recurrent problem for the image segmentation exists: a same kind of primitive can be represented on an image through pixels with different manners. Thus, primitives are difficult to detect using a single operator, or an operator which has the same settings when it is applied on the entire image. Primitives representation mainly depends on the image resolution (i.e. thickness of edges, size of regions), and image/acquisition quality (i.e. clear or blurred edges, noised regions). Operators are image processing algorithms applied locally around a pixel $P$. According to its settings, an operator affects more or less pixels surrounding $P$, which defines the location where this operator

[1] We consider a picture as a 2D matrix characterized by its width, its height and the information stored on each pixels. Here, this information represents radiometric values, provided by a scan device.

is applied. To solve this problem of representation variability, we need to locally adapt interpretations on the image, and not globally through only one algorithm applied on the entire image. In the scope of this paper, we focus on this issue by using a MAS approach, and we only consider the edge detection.

Usually, an edge is characterized by an abrupt gray level change between two regions. However, edges may be blurred: the gray level change is not abrupt but spread out through a lot of pixels. As for region detection, edge detection depends on how the used algorithm is tuned: some settings detect straight visible edges, while others detect partial edges. So, the difficulty is to define algorithms which are able to detect both partial and correct edges, without being disturbed by noise. Another difficulty is to manage hidden edges in order to then detect closed edges.

Multi-agent systems (MASs) are an interesting solution since they allow the cohabitation of several algorithms. For example, agents can analyze problems they are locally confronted with, and then use the algorithm which seems the more suited to their local context [1, 2, 11, 14].

As we will se later on, within the image processing domain, MASs are used according to two opposed software engineering strategies: 1) they can be used to exploit different macro level results provided by image processing algorithms, using negotiation among the agents [1, 2, 7, 11, 12, 14], or 2) they can be used as artificial social systems, composed of entities which exploit micro level image processing results [8], adapted to the intrinsic image processing issue we exposed previously.

Instead of conceptualizing macro level solutions that do not solve the problem of primitives representation variability, we propose to solve the image processing issues at the micro level. MASs offer this possibility thanks to the use of reactive autonomous entities that directly operate at the micro level.

We advocate that the micro approach can more successfully benefit from the interesting characteristics of MASs. Indeed, in such an approach, the idea is to use the different image processing algorithms as concrete perception and action tools for defining autonomous agents that will locally interact among themselves and with the environment (the image) at the micro level. A global behavior will then emerge from these micro level interactions. Our goal is edge detection within Scan images (See Figure 1).

Figure 1: 2D picture of a patient representing his lungs.

This paper is organized as follows. In Section 2, we study contributions that already use MAS for image processing. In Section 3, we give an overview of our multi-agent edge detection approach. In Section 4, we present the different algorithms of image processing we used. We then present in Section 5 our MAS model and the various agents behaviors. We end up by discussing our results and by giving future works.

## 2 Related work

Several works associating MAS and image processing already exist [1, 2, 7, 8, 11, 12, 14]. Different MAS properties and mechanisms have been used in these approaches (cooperation, negotiation, adaptation, emergence, etc...).

**Cooperative approaches** enable agents, each associated with an image processing algorithm, to share different collected information. For example, Settache et al. [12] use cooperation among agents to share the result of both region and contour segmentation algorithms, increasing the quality of the MRI[2] brain part's detection. More precisely, a quad tree algorithm is used to identify primitive regions, and a Shen filter is used to determine edges. The solution consists in merging regions according to their homogeneity. An edge agent can prohibit the fusion between two regions if it is located in between. Cooperation enables the cohabitation of several image processing algorithms. Its increases the quality of the segmentation by confronting the information provided by the different algorithms.

Abchiche et al. [1] developed a MAS which enables the emergence of concepts from the cooperation between agents. The proposed MAS is able to learn from its own experiences. It is an evolution comparing with the previous approaches where cooperation is predefined. Here, each agent is associated with one or more operators (image processing algorithms). An agent knows which operators it needs to apply, and what is the expected result of its application on the environment [3]. At the beginning, acquaintances evolve randomly, but as the cooperation process progresses, each agent knows the utility it can have to help its neighbors to accomplish their goal. Groups of agents *emerge*, so does the cooperation between the agents within each group. These groups represent the best solution to satisfy at the same time each member goal. In comparison with genetic algorithms, this MAS provides a distributed fitness function thanks to each agent goal. This is a real advantage because fitness functions are difficult to establish when the system has independent goals. However, as underlined by the author, the developed system does not contain any feedback function: the global behavior of a group, does not change the agents behavior to produce a new global behavior.

**Negotiation** between agents on different representations or interpretations is another advantage that MAS may provide. For example, Yanai [14] developed a MAS where the agents are used to structure the perceptions of the system. The proposed MAS recognizes objects on a real 3D scene. The system extracts primitive information like lines (Hough transform), edges (Snakes) or regions using different types of algorithms. Each agent is located on the image and builds a set of coherent primitives called representation. This representation is compared with a database to identify a potential object in the scene. Then, the agents interact to negotiate their local representations. Communications between agents make their representations evolve, in order to find common coherent representations. The system reaches its goal (objects recognition) when all the agents agree on their representations.

Mazouzi et al. [8] uses an **adaptive MAS** that enables the **emergence** of edges detection on pictures representing 3D scenes. The solution consists in defining the segmentation process using the already existing segmentation. So, this concretely applies a systemic loop which is a characteristic of adaptive systems. Thanks to the auto-organization property of the system, and without specific/complex image processing method, the system is able to make emerging contour detection. The solution cannot be compared with solutions that already exist, because this kind of approaches is not very mature. The MAS paradigm is better used in this approach, because the possible configurations number of the system is higher than with a system which uses macro image processing results. Macro image processing results limit the number of the system configurations/solutions. Then, it is possible to approach the best solution with a system based on micro image processing results.

To conclude, multi-agent system can be used through several ways. We have seen in [12, 7], that MAS are used to enable the cohabitation of several image processing methods, thanks to the MAS software engineering properties. These approaches give good results, but the benefit that comes

[2]MRI : Magnetic Resonance Imaging

from the use of the MAS paradigm is limited by the used algorithms that only provide results at the macro level. In [1], an efficient use of an emergent process is done enabling the system to learn from its own experiences. However, between these approaches, there is no work that deals with the intrinsic image processings problems as we explained them on the first paragraph of the introduction. It can be interesting to focus on defining systems that are adapted to the inherent problems of the image processing, and not adapted to already existing algorithms that limit the agent autonomy, and reduce the possibility of implementing self-organized systems with embedded emergent processes.

Defining such a process has been done in [8], where an adaptive system, with lightweight agents, has been developed to enable the edge detection without using any specific algorithm. However it seems that data representing detected edges cannot be used at a higher level of abstraction for defining a more complex system. So, the data representation has to be reified to make it more usable. We propose a system essentially constituted of reactive agents. The global behavior has to detect edges: we focused on a system conceptualization adapted to the edge detection problem. The data representation is reified so that it can be used in future and more complex systems.

# 3 Overview of our approach

The application domain that initially motivated this work is the automation of lung detection on images provided by a Scan device. This device send X rays through a body and interpolates the lost energy of these rays to construct a 3D image. In the scope of this paper, our approach is focused on 2D images.

Like for all the organs of the body, the problem is that the lungs shapes vary according the scaned body. As a first approach, we developed a system that is able to build a representation of the edges which are present on an image.

Traditional image processing algorithms, using filters of derivation like Kirsch, Shen or Canny, aims to represent contours present onto a source image into a destination image. These detected contours are represented thanks to grayscale values. High grayscales values on a located pixel means that there is a high probability that this pixel represents an edge. However, we must ponderate this interpretation, considering that noise can be present on the source image, and then that it can be created false edge representation on the destination image. Moreover, the body's organs consistence does not present right visible edges on the destinations images. The we must consider that some edges will not be detected by image processing filters.

Our work goal aims to construct an MAS model which interpret information given by conventional image processing algorithms to reconstruct and reorganize them. The MAS model developed on this paper aims to reconstruct an image's edges representation given by an image processing filter (We used a Kirsch filter on the section 4.2). This representation is computed through an MAS containing different agents, having different perceptions. First, the system initializes a predefined number of *exploration agents*. Exploration agents look for edges according to a gradient field that attracts them towards existing edges: the GVF (See section 4.3). They have a more precise perception of edges through the Kirsch filter which enables them to have a precise edge detection between two pixels. When they find an edge, *Edge following agents* have to follow the detected edges. They have to rebuild these edges by approximating them with segments. A segment is represented by two *node agents*, which are located on its extremities. It can be linked with other segments. A node agent which is not linked with another node agent is called an *end edge agent*. This last kind of agent has to negotiate with others end edge agents the closing of the edge it is representing. These negotiations are part of the reinterpretation by the system of the information provided by the image processing algorithms. The local work of all these agents makes emerging the edges representation through segments (see Figure 2).

The approximation of the segments depends on the user desire to have a precise or coarse edge representation. This gives the advantage to reduce a 2D shape into a 1D signal (the contour). Moreover, the precision of this signal can vary and edges can be represented through a multi-resolution data. This data could be analyzed by a more complex system that should be able to detect fuzzy shapes like lungs.

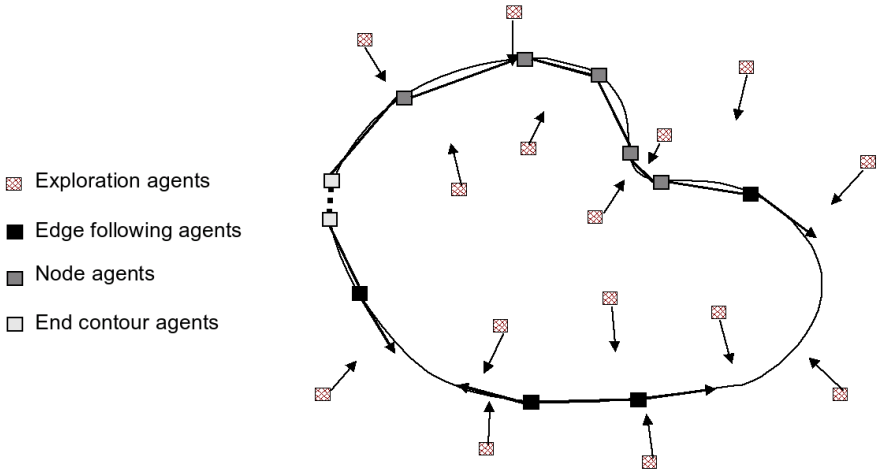


Figure 2: Overview of the edge detection.

# 4 Image processing's tools

Agents have a local perception of the gradient which enables them to look for edges randomly (see Section 4.2). To enrich this environment, we use a GVF[3] (see Section 4.3). A GVF is a field emitted in the environment that informs, at a given position, the nearest/highest surrounding gradient. It thus enables the agents to limit the randomness of their movements. To increase results quality of these tools (gradient perception and GVF), we have applied some pre-processing on the image. So, before describing our multi-agent approach, we describe in this section the image processing tools which are used to do the pre-processing tasks.

## 4.1 Pre-processing

To clean the image, we used a *Nagao* [4] filter which has the particularity to reduce the noise without destroying information like edges. Such a pre-processing is very important because it largely influences the result quality of the used algorithms. The Nagao filter provides satisfying results for our approach.

We have also defined a process that deletes non relevant information, that is, black areas surrounding the body. This process has no effect on the final result but decreases the computing time, especially the GVF computing time.

## 4.2 Gradient computing

An edge is characterized by a high grayscale variation. To measure this variation, we have to compute the gradient of the image pixels. In one pixel, a gradient is a vector defined by its amplitude and its direction. The amplitude directly relies on the local grayscale variation. The direction is orthogonal to the border. A high gradient corresponds to significant edge. A low gradient may correspond to a weak visible edge or simply noise.

In some cases, the gradient amplitude may be sufficient. In our case, the GVF computing requires to compute the gradient with the highest possible precision. A bad gradient, that does not detect diagonal edges, gives a corrupted GVF, or even a completely foolish GVF. So, we have chosen the Kirsch filter, and we have improved it. It enables to get a gradient with both good amplitude and good direction.

The GVF computing (see Section 4.3), needs to have a good gradient filter, but it also needs to have opposed vectors around an edge. The improved Kirsch filter enables us to obtain vectors inversely directed. Indeed their direction is as orthogonal as possible, considering the eight possible directions provided by the filter. Moreover, this filter enables to detect edges with a two pixels

[3]GVF : Gradient Vector Flow

thickness. The external edge is defined by a positive or negative value, whereas the internal edge has an opposed value (see Figure 3). This enables us to precisely detect the edge, between these two edges. Agents thus determine the edge thanks to the presence of these two internal and external edges. If opposed values are not present, agents do not consider them as an edge.

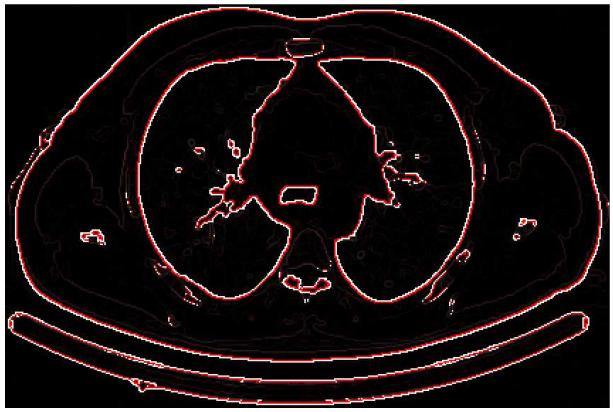

Figure 3: Gradient view of Figure 1. Red edges represent negative gradient, whereas white edges positive gradient. For more visibility, we increased the gradients.

The Kirsch derivative is the best filter that we found and its use has a direct influence on the GVF quality. However, gradients are computed onto a few number of pixels (3 pixels). This means that partial edges (as we described them in the introduction) are not detected. This problem is solved thanks to our multi-agent system.

### 4.3 Gradient Vector Flow (GVF)

Initially, the GVF [10] was developed to solve problems encountered by the snakes which are deformable edges. These limitations were mainly due to problems related to the snake initialization and its poor convergence on the concave regions.However, lot of prjects [13, 9] have demonstrated the GVF efficiency.

The GVF proceeds in two steps for the field computing:

- the gradient computing (See the previous section). We define $f(x, y)$ as the external snake energy : $f(x, y) = E_{ext}(x, y)$ and $E_{ext}(x, y) = |\overrightarrow{\nabla I(x, y)}|$.
- the gradient vector flow computing. It is defined as a field of vectors $\overrightarrow{\mathbf{v}(x, y)} = (u(x, y), v(x, y)$ which minimize the following energy:

$$\varepsilon = \iint \mu(u_x^2 + u_y^2 + v_x^2 + v_y^2) + |\overrightarrow{\nabla f}|^2 |\overrightarrow{\mathbf{v}} - \overrightarrow{\nabla f}|^2 dxdy$$

$\nabla$ is the Laplacien operator (second derivative); $f_x$ is the partial derivative of $f$ according $x$; $f_y$ is the partial derivative of $f$ according $y$; and $\mu$ is a coefficient which defines how a gradient can be dispersed in its vicinity. This is this coefficient which determines the perception field of the agent.

So, we can see that the GVF needs to compute a second derivative. The Kirsch derivative is a first derivative. However, because of the opposed computed vectors, the result is comparable with that obtained with a second derivative. The GVF algorithm gives an interesting result (see Figure 4), but it needs a lot of time to converge.

We do not make a cohabitation of this algorithm result. We just complete the environment by computing a field which represents potential edges. This enables to improve the convergence of the system behavior. Indeed, the Kirsch filter is a simple filter which enables the agents to locally determine an edge with precision. So, this result is related to the micro level.

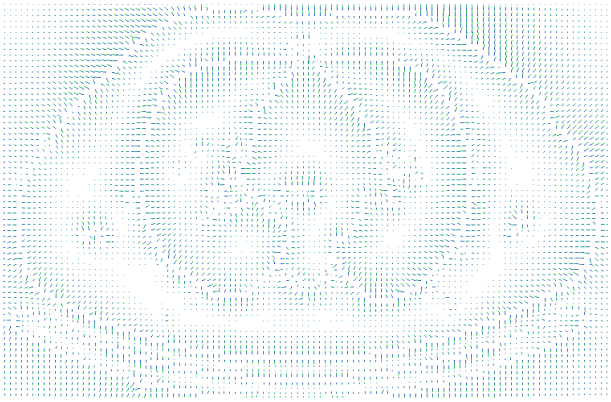

Figure 4: Result of the GVF computing (10000 iterations). This is a part of the agent's representation.

# 5 Multi-agent approach for the edge detection

## 5.1 Multi-agent model

The used MAS model is constituted by the agents and their environment. The environment contains the image. Each pixel of this image characterizes a gray level (the radiometric information provided by the scan) and contains a boolean value which defines if the pixel has already been explored by an agent. On the they are located, agents locally perceive the gradient which defines a right visible edge, and the GVF which gives an "average direction" between the nearest and the highest potential edge (Figure 5).

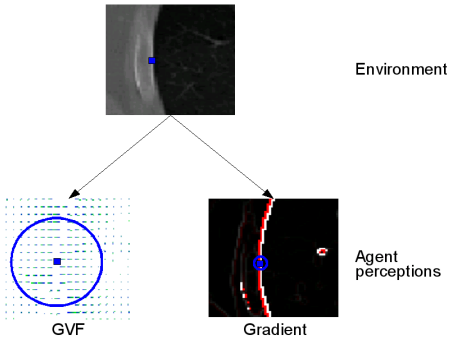


Figure 5: Two different perceptions of a single image. The blue square represents an agent. The circle surrounding this agent represents its perception field. The GVF perception is wider than the gradient perception.

Agents have different behaviors according to their current state and perception:

1. Exploring the environment looking for an edge.
2. Following a detected edge.
3. Closing an edge by negotiating with other agents.

## 5.2 Exploration behavior

The exploration behavior consists in looking for an edge. For this, the agent moves according to the GVF and a random direction as follows:

$$P_{a_{t+\triangle t}} = P_{a_t} + V_d \triangle t \overrightarrow{D_{a_{t+\triangle t}}}$$

$P_a$ represents the agent position; $\overrightarrow{D_{a_{t+\triangle t}}}$ represents the agent direction; $V_d$ represents the movement speed of the agent in pixel per milliseconds. By default, $V_d = 0.02\ pixels/ms$; $\triangle t$ represents the time variation in milliseconds since the previous iteration. $\triangle t = t_{\text{current iteration}} - t_{\text{previous iteration}}$.

The agent direction is defined as follows:

$$\overrightarrow{D_{a_{t+\triangle t}}} = P_h \overrightarrow{D_{h_{t+\triangle t}}} - (1 - P_h)\overrightarrow{D_{GVF}(P_{a_t})}$$

$\overrightarrow{D_h}$ represents the chaotic direction; $\overrightarrow{D_{GVF}(X)}$ represents the GVF vector interpolated at the X position, and inversely directed to the nearest/more significant local gradient. $P_h$ represents the percentage of taking account of the chaotic component. If $P_h$ is equal to $O\%$, then the movement of the agent will be determined only through the GVF. Inversely, if $P_h$ is equal to 100%, the agent will move randomly and will not consider the GVF. By default, $P_h$ is equal to 20%.

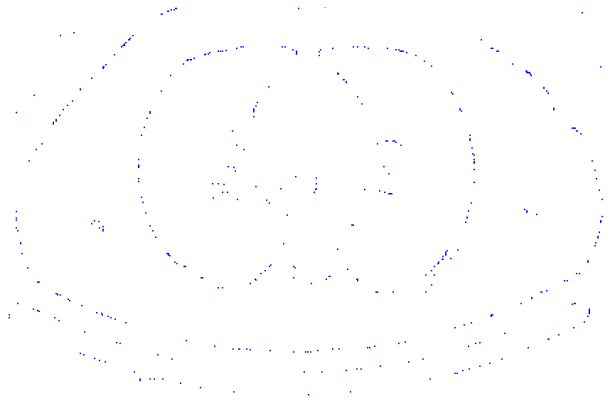

Figure 6: Result obtained using only the exploration behavior.

Figure 6 present the result of using only the exploration behavior, behavior which can be named as "follow the GVF" and "move randomly". It is easy to guess the lung shape represented by the MAS. In other words, without any other behavior, a shape has already emerged. But the exploration is not completely described.

The exploration behavior is divided in three steps:

1. the agent checks if it is on a potential edge, not already visited by another agent. If it is the case, the agent determines the various possible directions of the edge (See Figure. 7). It creates edge following agents for the detected directions. The created agents will follow the edge (see Section 5.3). After having detected an edge, the exploration agent changes into a node agent or disappears (see Figure 8).

2. the agent moves to a potential gradient thanks to the GVF. It computes its new position $P_{a_{t+\triangle t}}$ as expressed before. In some places of the GVF, the direction is equal to $\overrightarrow{0}$. So the agent follows a random direction. Agents can thus go to edges which do not express gradients sufficiently high to appear in the GVF.

3. finally, if the exploration agent does not find any edge before a deadline, it disappears.

## 5.3 Edge following behavior

Edge following agents are created by exploration agents which have found an edge. They have to reconstruct the edge by representing it with a succession of segments. An edge following agent is linked with another one which explores the same edge in the opposite direction (see Figure 8), or simply with a node agent which does not move. To construct the edge, it moves pixel per pixel. Progressively, it moves away from its neighbor agent. In each step, it stores the pixel position

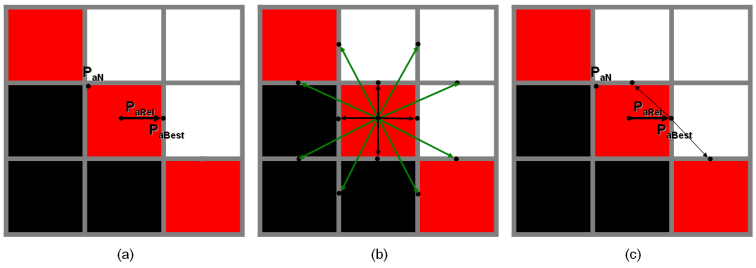


Figure 7: **representation of 3x3 pixels located according the agent position $P_{aN}$.** The agent can determine with precision the direction to follow onto an edge. In (a), the agent computes the best position $P_{aBest}$ according the surrounding edge. $P_{aRel}$ is deduced from $P_{aBest}$. In (b), we can see all possible directions: near directions in black, and far direction in green. In (c), the agent has chosen two directions.

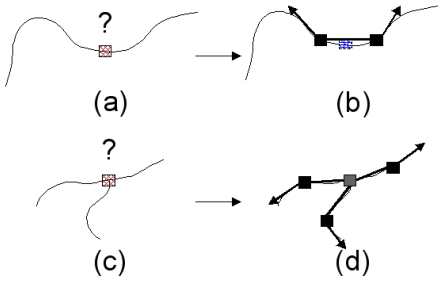


Figure 8: On (a), the exploration agent finds an edge. It determines two possible directions to follow on the edge. Then in (b), it creates two edge following agents. In (c), the exploration agent finds a node linking three edges. Then, it creates three edge following agents in (d), and transforms itself into a node agent.

located between the two edges that it finds (see the section 4.2). The objective is to make sure that each position characterizing the edge portion which separates it from its neighbor, can be approximated (using a threshold) by the segment connecting the two agents. In other words, as soon as the list of the concerned positions does not form a line any more, the agent stops (see the segment pertinence test part). Then another agent continues its work. By varying the threshold, the precision of the detection can be tuned.

The follow-up of an edge, all the edge behavior includes several steps:

1. attempting a fusion: starting from the position where it is, the edge agent checks if there are directions to follow, already exploited by other agents. If it is the case, it asks a fusion with these agents (Cf. Figure 9) and disappears.

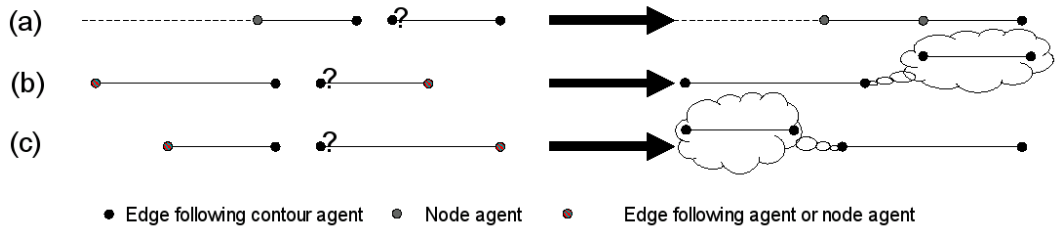


Figure 9: Fusion of two edge agents. Three scenarios are possible: a,b and c depending on the status of their neighbors. An agent which is linked to a node agent has priority, then the length of the segment prevails.

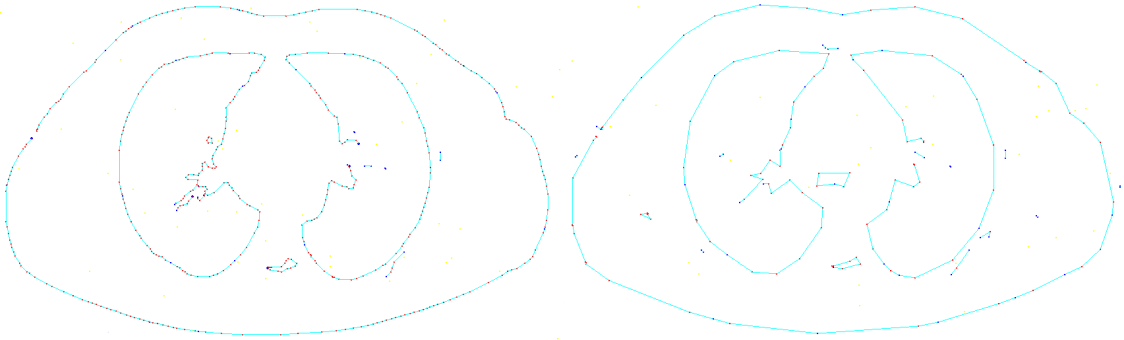

Figure 10: Contour detection using our MAS. On the left side, we have precise contours. On the right side, we got more coarse contours.

2. edge following: always starting from its position, it establishes all the directions not exploited to follow:
   - if there is no possible direction, the agent becomes an end edge agent.
   - if the agent has only one direction to follow, it checks if the segment formed between the position of its neighbor agent and its position, is an acceptable segment, i.e. if it is a segment which makes a good interpolation of the edge part. If this segment is valid, then the agent stores its position in its list of positions and advances. Otherwise, it stops its behavior, becomes a node agent, and creates a new edge following agent which will try in its turn, to have the longest segment.
   - if it has more than one direction to follow, the agent becomes a node agent connected with several directions. It creates as many agents of edge as detected directions. These agents will continue to follow the considered edge.

### 5.4 Closing edge behavior

The closing edge behavior is associated with end edge agents. An end edge agent aims to be connected with the nearest end edge agent. Nevertheless it is connected only if the distance between itself and this nearest agent is lower than a threshold. The link established between the two agents is noted as being a link of speculation. Thus, if another end edge agent appears later, and if it is nearest than the first detected end edge agent, the link of speculation is replaced. That implies that an end edge agent always keeps its closing edge behavior, in order to always have the capability to get the better configuration. It also keeps its node behavior because it is connected with more than one agent.

## 6 Results

We tested the system on several Scan images representing lungs. The detection of contours through our MAS is very effective (see Figure. 10). The contour approximation is done correctly. It can be very precise or very coarse. However, it may happen that two agents meet to form a node agent.This node agent is connected with two collinear segments which should form one segment. This part should be improved.

We have also tested the same images disturbed by noise. Results are interesting because a high number of edges are detected. However, a more precise closing contour behavior is required in such a context. Such an improvement would be very effective to detect edges on highly disturbed images.

When activating only exploration behavior, we observe that the agents converge very quickly on edges. So, the GVF appears to be very useful. But, by activating only the exploration agents behaviors and just by observing their moves, some parts are not visited by the agents. That is due to the fact that the exploration agents cannot leave the contour which "captured them" and then detect weaker edges which are less perceptible through the GVF. Then, the contribution of

the chaotic aspect during the movement of the agents is very important, because it precisely makes possible for an agent to escape from a contour, and thus to explore edges which are less perceptible through the GVF. If the GVF is completely inactive, the edges detection is longer, because we need more agents. However, if we take into account the computation of the GVF which is hopelessly long, the alone use of the chaotic component is definitely faster.

The quality of the edge reconstruction through our MAS, depends directly with the quality of the used image processing algorithms. It depends also of how the system treat information provided by image processing filters. To measure this quality, we must analyze what information detected by the Kirsch filter are detected by our MAS. In fact, this depends on the image treated. If the image contains lot of edges, the system must be initialized with lot of exploration agents. If the number of agents is not sufficiently high, some edges will not be detected. This is due to the fact that exploration agents which find an edge are replaced by edge following agents. When all exploration agents have found an edge, the system is unable to detect isolated edges. To correct this problem, exploration agents do not have to die when they have found an edge. They must continue their work. An exploration agent must die when it has not found any edge during a defined time.

# 7 Conclusion

We have presented in this paper a multi-agent approach capable of interpreting information provided by an edge detector (the Kirsch filter). This system is able to represent edges present on a 2D picture through segments. Lacking edges are reconstructed thanks to protocols of negotiations between agents which represent end edges. These segments are represented by two node agents located on their extremeties. They approximate edges with a more or less precision according to the user's requirements. This user can be a human user or a more complex system able to recognize itself present objects on the picture.

We enriched the environment by adding a field (GVF) which enables agents to feel locally the nearest/highest edges. The GVF enables to have faster convergence of the system. It also helps with better detecting edges, and avoids agents to loose themselves into regions which lack of information. However, considering the high preprocessing time of the GVF, the contribution given by this method is finally moderated. Considering image processing tools, the Kirsch derivative is more interesting because it is a very simple image processing algorithm that really improves the local perception of the agents at the micro level. So, we are convinced that a very efficient system could be built using a full micro level approach, applying MAS fundamental mechanisms such as stigmergy for instance. Such an approach will overcome the classic image processing algorithms that are limited to macro results which can not take into account the local characteristics of a complex image.

# 8 Future works

In order to detect edges, we have used a Kirsch filter that we improved. As perspectives, it can be interesting to enrich the agent perception by giving them edges information through others image processing algorithms more specialized with specific edge detections. We can also approximate edges thanks to curved primitives, like B-Spline, instead of segments. The closing edge behavior can be improved by adding a second threshold for the detection of potential gradients in order to take into account edges which are less visible. For example, when an end edge has been detected, local exploration agents can be sent to locally explore the image with a better sensitivity.

More generally, the system is able to treat information provided by image processing algorithms. Then two questions can be asked to improve the MAS: is the system able to take into account the limitations/defects of each used algorithm? By answering positively to the previous question, we can deduce that the system is able to "understand" algorithms it is using. So, is the system able to improve these algorithms, and is it able to create new algorithms according its needs?

This part of research induce the study of an image processing model used to construct an environment whose atomic conceptual elements are relied with an extreme coherence.

# References


[1] Yazid Abchiche, Patrice Dalle, and Yohann Magnien. Construction adaptative de concepts par structuration d'entités de traitement d'images. In *RFIA 2002*, pages 1043–1051, Angers, January 2002. AFRIF-AFIA.

[2] Ernst G. P. Bovenkamp, Jouke Dijkstra, Johan G. Bosch, and Johan H. C. Reiber. Multi-agent segmentation of IVUS images. *Pattern Recognition*, 37(4):647–663, 2004.

[3] Philippe Dejean and Patrice Dalle. Image Analysis Operators as Concept Constructors . In *IEEE Southwest Symposium on Image Analysis and Interpretation , San Antonio, ÃL'tats-Unis*, pages 66–70, April 1996. Dates de confÃl'rence : avril 1996.

[4] Didier Demigny, Jean Devars, Lounis Kessal, and Jean FranÃğois Quesne. Implantations temps réel du filtre de lissage d'images de nagao. In *Traitement du Signal*, volume 10, 1993.

[5] Jacques Ferber. *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*. Addison-Wesley Longman Publishing Co., Inc., 1999.

[6] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. Addison-Wesley, 2002.

[7] Radia Haroun, Fatima Boumghar, Salima Hassas, and Latifa Hamami. A Massive Multi-agent System for Brain MRI Segmentation. In *MMAS*, pages 174–186, 2004.

[8] SmaÃŕne. Mazouzi, Mohamed C. Batouche, and Zahia Guessoum. A self-adaptative multi-agent system for segmentation and reconstruction of 3d scenes. Luxembourg, November 2004. AISTA 2004 in cooperation with the IEEE Computer Society Proceedings.

[9] Nikos Paragios, Olivier Mellina-Gotardo, and Visvanathan Ramesh. Gradient vector flow fast geodesic active contours. pages 67–75.

[10] Jerry L. Prince and Chenyang Xu. Gradient Vector Flow : A New External Force Model for Snakes. In *IEEE Image and Multidimensional Signal Processing Workshop*, pages 30–31, 1996.

[11] Nathalie Richard, Michel Dojat, and Catherine Garbay. Dynamic Adaptation of Cooperative Agents for MRI Brain Scans Segmentation. In *AIME '01: Proceedings of the 8th Conference on AI in Medicine in Europe*, pages 349–358, London, UK, 2001. Springer-Verlag.

[12] Hakim Settache, Christine Porquet, and Su Ruan. Une plate-forme multi agents pour la segmentation d'images : application dans le domaine des IRM cérébrales 2D. Technical report, Université de Caen, 2002.

[13] Chenyang Xu, Jr. Anthony Yezzi, and Jerry L. Prince. On the relationship between parametric and geometric active contours. In *34th Asilomar Conference on Signals, Systems, and Computers*, pages 483–489, October 2000.

[14] Keiji Yanai. An image understanding system for various images based on multi-agent architecture, December 1999.