Combining geometrical and textured information to perform image classification

Jean-François Aujol & Tony F. Chan *

UCLA, Mathematics Departement
405 Hilgard Avenue, Los Angeles, California 90095-1555
email: {aujol,chan}@math.ucla.edu

9th November 2004

Abstract

In this paper, we propose a framework to carry out supervised classification of images containing both textured and non textured areas. Our approach is based on active contours. The classification corresponds to the minimum of a functional. Using a decomposition algorithm inspired by the recent work of Y. Meyer, we can get two channels from the original image to classify: one containing the geometrical information, and the other the texture. Using the logic framework by Chan and Sandberg, we can then combine the information from both channels. Thus, we design a classification algorithm in which the different classes are characterized both from geometrical and textured features. Since natural images are combinations of both textured and non textured patterns, this new approach considerably enlarges the scope of possible applications for active contours-based classification algorithms.

Key-words: Classification, texture, geometrical image, decomposition, logic model, level-set, active contour, PDE, wavelets.

1. Introduction

The problem of image classification consists in assigning a label to each pixel of an image. This label indicates to which class belongs a pixel. This is one of the basic problem in image processing. The classification problem is closely related to the segmentation one, in the sense that the aim is to get a homogeneous partition of the image. In the classification problem, each region represents a class. Many classification models have been developed: region growing algorithms have been used [21, 24, 41], as well as stochastic approaches [12, 13, 23, 25, 22], and most recently variational approaches [5, 33, 34, 31, 40, 8].

*Both authors were supported by grants from the NSF under contracts DMS-9973341, ACI-0072112, INT-0072863, the ONR under contract N00014-03-1-0888, the NIH under contract P20 MH65166, and the NIH Roadmap Initiative for Bioinformatics and Computational Biology U54 RR021813 funded by the NCRR, NCBC, and NIGMS.

The approach which we use here is inspired from [33, 44, 8, 36], and is based on active contours [5, 14, 39, 29, 45, 17, 42]. The partition we seek is a minimizer of a functional. We compute this minimizer by solving the associated PDE's system. These PDEs guide the interfaces (zero level sets) towards the boundary of the optimal partition thanks to internal (regularity of the interface) and external (data term and partition) forces.

The kind of approach we use [33, 44, 7] has the advantage to be faster than the statistical one, whereas the results we get are at least as good (as long as the images to classify are not too noisy) [35]. Through active contours and PDEs, it is easy to introduce regularization constraints on the geometrical shape of the classification regions.

But, as far as we know, none of the existing classification models using a variational approach can deal with images containing both textured and non-textured areas. For instance, [33, 44, 17] are designed to work with non-textures images whereas [31, 37, 8] are specifically devoted to textured images classification.

The idea of our method is as followed. Given an original image, we first split it into a geometrical component and a textured component thanks to the algorithm of [7, 6] (in the case when the original image has been corrupted by some additive noise, we use the decomposition algorithm of [9]). We thus get two channels from our original image: a geometrical one, and a textured one. We construct a data term associated to the geometrical (resp. textured) component by using the method of [33] (resp. [8]). We combine them thanks to the logical framework introduced in [36], and we are then in position to carry out the classification of our original image containing both textured and non-textured areas. One should remark that, since we use the logical framework of [36], a class can be characterized both by textured and non-textured features.

The use of a decomposition algorithm to split an image into a geometrical component and a textured component, in order to apply adapted algorithms to each part of the image, has already been introduced successfully in image inpainting [11] and in image nonlinear interpolation [10].

The plan of the paper is as followed. In Section 2, we first introduce the decomposition models of [7, 6, 9]. In Section 3, we present a general classification framework as used in [33, 8]. We emphasize on the choice of the data term in Section 4. In Section 5, we recall the logic model of [36] and show how we can use it in our classification algorithm. We illustrate our approach with some numerical examples.

2. Decomposition model

In [26], Meyer discussed the classical Rudin-Osher-Fatemi model [32]. He introduced a new model to split a given image f into a sum $u + v$ of a bounded variation component and a component containing the oscillating part of the image. The u component can be viewed as a sketch of the original image f. This model has first been successfully implemented by Vese and Osher [43]. A different approach has been proposed in [7, 6]. We will use this one, where the decomposition is computed by minimizing a convex functional which depends on the two variables u and v, alternatively in each variable. Each minimization is based on a projection algorithm to minimize the total variation [15]. See also [30, 9, 38, 20] for other interesting approaches.

2.1 Modeling

Let Ω be a bounded open set of $\mathbb{R}^2$ with Lipschitz boundary. The space used to model the geometrical component u of an image f is the space BV of functions with bounded variation. A function f belongs to $BV(\Omega)$ if it is in $L^1(\Omega)$, and if its total variation is finite. We will denote by $J(f)$ the total variation of f. Formally, we have:

$$J(u) = \int_\Omega |\nabla u| \tag{2.1}$$

We refer to [1, 15, 26, 5] for a deeper insight in BV.

In [26], Meyer has proposed a new decomposition model:

$$\inf_{(u,v)\in BV\times G/f=u+v} (J(u) + \alpha\|v\|_G) \tag{2.2}$$

The Banach space G contains signals with large oscillations, and thus in particular textures and noise. The G-norm replaces the classical L^2 norm of the Rudin-Osher-Fatemi model [32]. We give here the definition of G:

Definition 2.1. *G is the Banach space composed of the distributions f which can be written*

$$f = \partial_1 g_1 + \partial_2 g_2 = \operatorname{div}(g) \tag{2.3}$$

with g_1 and g_2 in L^∞. On G, the following norm is defined:

$$\begin{aligned} \|v\|_G \;=\; & \inf\{\|g\|_{L^\infty} = \; /v = \operatorname{div}(g), \\ & g = (g_1, g_2),\ g_1 \in L^\infty, g_2 \in L^\infty, \\ & |g(x)| = \sqrt{|g_1|^2 + |g_2|^2}(x)\Big\} \end{aligned} \tag{2.4}$$

A function belonging to G may have large oscillations and nevertheless have a small norm [26, 9, 3]. For instance, consider the sequence of functions f_n defined on $\Omega = (-\pi, \pi)^2$ by:

$$f_n(x,y) = \cos(nx) + \cos(ny) = \operatorname{div}\left(\frac{1}{n}(\sin(nx), \sin(ny))\right) \tag{2.5}$$

It is easy to check (see [3] for details) that $\|f_n\|_{L^\infty} = 1$ and $\|f_n\|_{L^2(\Omega)} = 2\pi$, but $\|f_n\|_G \leq \frac{1}{n}$.

2.2 Functional

In [7, 6], the authors have introduced the following functional:

$$\inf_{(u,v)\in BV\times\mu B_G} \left(J(u) + \frac{1}{2\lambda}\|f-u-v\|_{L^2}^2\right) \tag{2.6}$$

where

$$\mu B_G = \{v \in G(\Omega)/\|v\|_G \leq \mu\} \tag{2.7}$$

The parameter λ controls the L^2-norm of the residual $f-u-v$. The smaller it is, the smaller the L^2-norm of the residual gets. And μ controls the G-norm of the oscillating part v. It is shown in [7, 6] that solving (2.6) is a way to solve (2.2): indeed, when λ goes to zero,

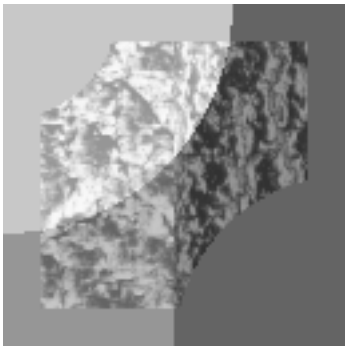

Figure 1: Original Synthetic image

this means $f = u + v$ in (2.6), and the parameter α in (2.2) is just the Lagrange multiplier associated to the constraint $v \in G_\mu$.

The minimum of (2.6) is computed by minimizing alternatively in each variable u and v. Each minimization is based on a projection algorithm to minimize the total variation [15].

We denote by P_K the orthogonal projection on a set K. When $K = \mu B_G$ for some $\mu > 0$, this projection is computed thanks to Chambolle's algorithm [15]. Notice that this is not a linear projection. The algorithm of [15] is based on a fixed point method.

Algorithm:

1. Initialization:
$$u_0 = v_0 = 0 \tag{2.8}$$

2. Iterations:
$$v_{n+1} = P_{\mu B_G}(f - u_n) \tag{2.9}$$
(this amounts to minimizing (2.6) with respect to v when u is fixed)

$$u_{n+1} = f - v_{n+1} - P_{\lambda B_G}(f - v_{n+1}) \tag{2.10}$$
(this amounts to minimizing (2.6) with respect to u when v is fixed)

3. Stopping test: we stop if
$$\max(|u_{n+1} - u_n|, |v_{n+1} - v_n|) \leq \epsilon \tag{2.11}$$

We show the efficiency of this algorithm by using it on the image of Figure 1. The obtained decomposition is displaid on Figure 2.

2.3 Noisy case

When the original image is corrupted by some additive Gaussian noise, one needs to modify the previous decomposition model so that it splits an image f into three components: a first component u containing the geometrical information, a second component v containing the textured information, and a last component w which is the noise. Such a decomposition algorithm has been developped in [9]. We recall here its principle. It relies on the use of a third functional analysis space for the noise component. We already have BV for the geometrical component and G for the textured one.

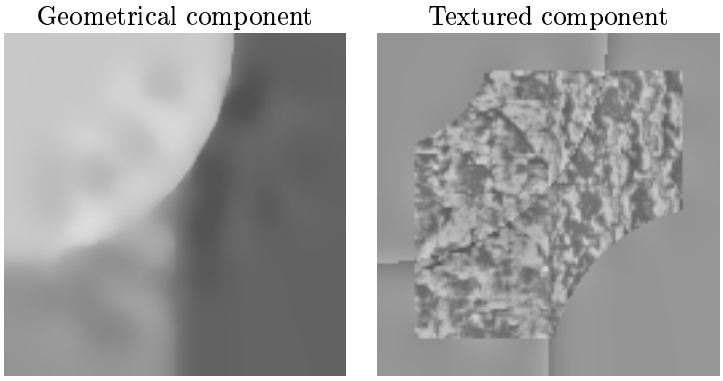

Figure 2: $u + v$ decomposition

Definition 2.2. $\dot{B}^1_{1,1}$ is the usual homogeneous Besov space (see [26]). Let $\psi_{j,k}$ an orthonormal base composed of smooth and compactly supported wavelets. $\dot{B}^1_{1,1}$ is a subspace of $L^2(\mathbb{R}^2)$, and a function f belongs to $\dot{B}^1_{1,1}$ if and only if: $\sum_{j\in\mathbb{Z}}\sum_{k\in\mathbb{Z}^2}|c_{j,k}|2^{j/2} < +\infty$, where $c_{j,k}$ are the wavelet coefficients of f.

Definition 2.3. The dual space of $\dot{B}^1_{1,1}$ is the Banach space $E = \dot{B}^\infty_{-1,\infty}$. It is characterized by the fact that the wavelet coefficients of a generalized function in $E = \dot{B}^\infty_{-1,\infty}$ belong to $l^\infty(\mathbb{Z}\times\mathbb{Z}^2)$.

It is this space, E, which is used in [9] to modelize the noise component.
Let us denote by

$$\delta B_E = B_{E(\delta)} = \{w \ / \ \|w\|_E \leq \delta\} \tag{2.12}$$

In [9], the authors compute the $u + v + w$ decomposition by minimizing the following functional:

$$\inf_{(u,v,w)\in BV\times\mu B_G\times\delta B_E}\left(J(u) + \frac{1}{2\lambda}\|f-u-v-w\|^2_{L^2}\right) \tag{2.13}$$

The minimum of (2.13) is computed by minimizing alternatively in each variable u, v and w. Each minimization is based on a projection algorithm to minimize the total variation [15], or on a wavelet soft thresholding [16]. The algorithm proposed in [9] is the following (we use the same notations as before):

Algorithm:

1. Initialization:
$$u_0 = v_0 = w_0 = 0 \tag{2.14}$$

2. Iterations:
$$\begin{aligned} w_{n+1} &= P_{\delta B_E}(f-u_n-v_n) \\ &= f-u_n-v_n-WST(f-u_n-v_n,\delta) \end{aligned} \tag{2.15}$$

where $WST(f-u_n-v_n,\delta)$ stands for the wavelet soft-thresholding of $f-u_n-v_n$ with threshold δ.

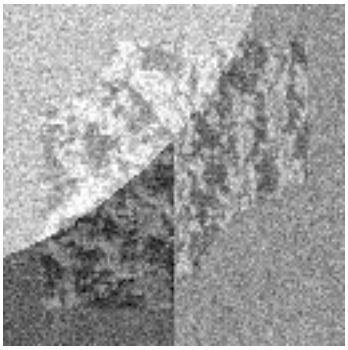

Figure 3: Noisy image ($\sigma = 20$)

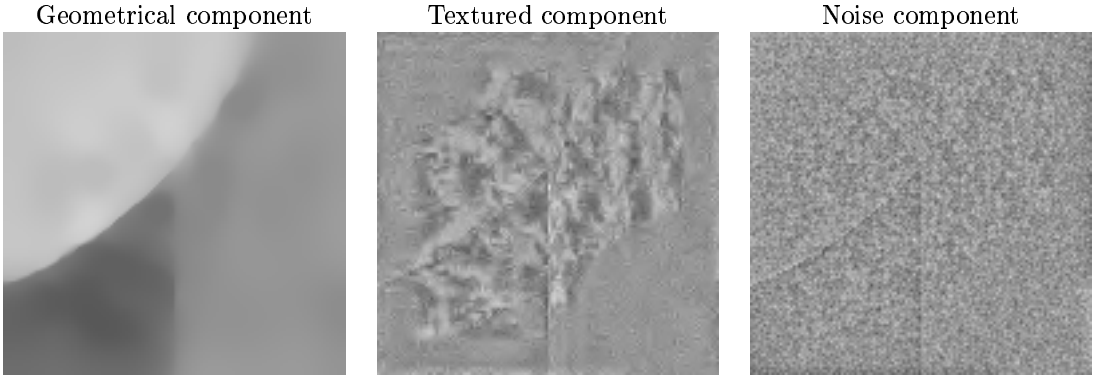

Figure 4: $u + v + w$ decomposition

$$v_{n+1} = P_{\mu B_G}(f - u_n - w_{n+1}) \tag{2.16}$$

$$u_{n+1} = f - v_{n+1} - w_{n+1} - P_{\lambda B_G}(f - v_{n+1} - w_{n+1}) \tag{2.17}$$

3. Stopping test: we stop if

$$\max(|u_{n+1} - u_n|, |v_{n+1} - v_n|, |w_{n+1} - w_n|) \leq \epsilon \tag{2.18}$$

We show the efficiency of this algorithm by using it on the image of Figure 3 (it is the picture of Figure 1 after it has been degraded by some additive Gaussian noise with standard deviation $\sigma = 20$). The obtained decomposition is displaid on Figure 4.

With the decomposition algorithms of [7, 6] and [9], we are thus able to isolate the geometrical component of an image, as well as its textured one. In the next section, we present a general classification framework which can be used on these two components [33, 8].

3 Classification framework

Our approach here is inspired from [33, 8]. We will use the same classification framework.

3.1 Partition, level set approach

The image is considered as a function $u_0 : \Omega \mapsto \mathbb{R}$ (where Ω is an open subset of $\mathbb{R}^2$). We denote by $Cl_k = \{x \in \Omega \,/\, \text{x belongs to the class k}\}$. In order to get a functional formulation rather than a set formulation, we assume that for all $k = 1 \dots K$, Cl_k is an open set Ω_k given by a Lipschitz function $\Phi_k : \Omega \to \mathbb{R}$ such that:

$$\begin{cases} \Phi_k(x) > 0 & \text{if } x \in \Omega_k \\ \Phi_k(x) = 0 & \text{if } x \in \partial\Omega_k \\ \Phi_k(x) < 0 & \text{otherwise} \end{cases}$$

(typically, Φ_k is the signed distance function to $\partial\Omega_k$, Φ_k being then lipschitz as soon as $\partial\Omega_k$ is lipschitz). Ω_k is thus completely determined by Φ_k (i.e. $x \in \Omega_k \Leftrightarrow H(\Phi_k(x)) = 1$, where H is the Heaviside function). The collection of open sets $\{\Omega_k\}$ forms a partition of Ω if and only if $\Omega = \bigcup_k \Omega_k \bigcup_k \partial\Omega_k$, and if $k \neq l$ $\Omega_k \bigcap \Omega_l = \emptyset$.

3.2 Regularization

We also use the Dirac distribution δ. In order that all the expressions we write have a mathematical meaning, we will use the classical regular approximations of these distributions.

$$\delta_\alpha(s) = \begin{cases} \frac{1}{2\alpha}\left(1 + \cos\frac{\pi s}{\alpha}\right) & \text{if } |s| \leq \alpha \\ 0 & \text{if } |s| < \alpha \end{cases}$$

$$H_\alpha(s) = \begin{cases} \frac{1}{2}\left(1 + \frac{s}{\alpha} + \frac{1}{\pi}\sin\frac{\pi s}{\alpha}\right) & \text{si } |s| \leq \alpha \\ 1 & \text{if } s > \alpha \\ 0 & \text{if } s < -\alpha \end{cases}$$

When $\alpha \to 0$, we have $\delta_\alpha \to \delta$ and $H_\alpha \to H$ (in the distributional sense).

3.3 Functional

Our functional has three terms:

1. $$F_\alpha^A(\Phi_1, \dots, \Phi_K) = \lambda \int_\Omega \left(\sum_{k=1}^K H_\alpha(\Phi_i) - 1\right)^2 \tag{3.19}$$

   Minimizing this energy term ensures that the result is indeed a partition of the image. It penalizes the pixels which are unclassified ($\sum_{k=1}^K H_\alpha(\Phi_k) = 0$), as well as the ones classified in at least two regions simultaneously ($\sum_{k=1}^K H_\alpha(\Phi_k) > 1$). In fact, to get a partition of Ω, we would need to have $\sum_{k=1}^K H(\Phi_k(x)) = 1$, $\forall x \in \Omega$. We rather minimize the quadratic error, which is easier.

   This type of energy term has first been introduced in [44]. A more elegant way to impose the partition constraint has been recently proposed in [42].

2. $$F^B(\Phi_1, \dots, \Phi_K) = \sum_{k=1}^K \gamma_k |\partial\Omega_k| \tag{3.20}$$

This term penalizes the contours length, which prevent from having too irregular contours and a lot of small regions. By using the co-area formula, it is possible to show that (the proof is given in [33]):

$$\lim_{\alpha \to 0} \int_{\Omega} \delta_{\alpha}(\Phi_k) |\nabla \Phi_k| dx = |\partial \Omega_k| \tag{3.21}$$

Then, in practice, we seek to minimize:

$$F_{\alpha}^{B}(\Phi_1, \ldots, \Phi_K) = \sum_{k=1}^{K} \gamma_k \int_{\Omega} \delta_{\alpha}(\Phi_k) |\nabla \Phi_k| \tag{3.22}$$

3.

$$F^{C}(\Phi_1, \ldots, \Phi_K) = \sum_{k=1}^{K} e_k \int_{\Omega} H_{\alpha}(\Phi_k) B_k \, dx \tag{3.23}$$

This last term stands for the data term, and we will explain how to construct it in the following sections.

Complete functional: The functional we want to minimize is the sum of the three previous terms:

$$\begin{aligned} F(\Phi_1, \ldots, \Phi_K) &= F^{A}(\Phi_1, \ldots, \Phi_K) + F^{B}(\Phi_1, \ldots, \Phi_K) \\ &\quad + F^{C}(\Phi_1, \ldots, \Phi_K) \end{aligned} \tag{3.24}$$

See [4] for a theoretical study of this functional in the two phases case.

3.4 Euler-Lagrange Equations

If $\Phi_1, \ldots, \Phi_K$ minimize the functional F, then necessarily (if furthermore the $\Phi_1, \ldots, \Phi_K$ verify some regularity conditions (see [5] for instance)), we have for $k = 1 \ldots K$: $\frac{\partial F}{\partial \Phi_k} = 0$. Assuming that Neumann conditions are verified, the associated Euler-Lagrange equations to F give the following system composed of K-coupled PDE's (see the appendix of [33] for instance): we have for $k = 1 \ldots K$

$$\begin{aligned} 0 &= \lambda \delta_{\alpha}(\Phi_k) \left(\sum_{q=1}^{K} H_{\alpha}(\Phi_q) - 1 \right) \\ -\gamma_k \delta_{\alpha}(\Phi_k) \text{div} \left(\frac{\nabla \Phi_k}{|\nabla \Phi_k|} \right) &+ e_k \delta_{\alpha}(\Phi_k) \left(\sum_{i=1}^{I} B_k^i(x) \right) \end{aligned} \tag{3.25}$$

3.5 Dynamical scheme

To solve the PDE's system (3.25), we embed it in the following dynamical scheme ($k = 1 \ldots K$):

$$\begin{aligned} \frac{\partial \Phi_k}{\partial t} = -\delta_{\alpha}(\Phi_k) \Bigg[& \lambda \left(\sum_{q=1}^{K} H_{\alpha}(\Phi_q) - 1 \right) \\ & -\gamma_k \text{div} \left(\frac{\nabla \Phi_k}{|\nabla \Phi_k|} \right) + e_k \left(\sum_{i=1}^{I} B_k^i(x) \right) \Bigg] \end{aligned} \tag{3.26}$$

where the initial condition $\Phi_k(0,x)$ is the Euclidean signed distance function to the zero level set Φ_k.

We discretize this system with finite differences (see [33]). We arbitrarily set $\alpha \in \mathbb{R}^+$ (in our experiments, we have used $\alpha = 3.0$). We need to set α small enough so that the approximations of the Dirac and Heaviside distributions (δ_α and H_α) remain reasonable. We also prefer α to be small since the bandwidth on which PDE's (3.26) are not empty depends on it (if $\alpha \in \mathbb{N}$, it contains $2\alpha - 1$ pixels width when ϕ_k is the signed distance to its zero level set). On the contrary, we need α not to be too small so that the band contains enough pixels to compute the different terms of (3.26).

3.6 Reinitialization

The Φ_k are initialized as Euclidean signed distance functions. Nevertheless, as in the classical active contour method [5]), the evolution of the Φ_k with respect to (3.26) does not keep them as Euclidean signed distance functions to their zero level sets. This prevents the convergence of (3.26) towards some Φ_k minimizing F. That is why it is necessary to periodically reinitialize the Φ_k functions into Euclidean signed distance functions.

To do so, we use the PDE:

$$\frac{\partial \Phi}{\partial t} + \text{sign}(\Phi_k)(|\nabla \Phi| - 1) = 0 \tag{3.27}$$

A theoretical study of this PDE, as well as an extension to more general Hamiltonians, can be found in [2].

3.7 Choice of parameters

Parameters: In our experiments, we always choose $e_1 = \cdots = e_K = 1.0$ and $\gamma_1 = \cdots = \gamma_K = \gamma$ ($\gamma \in \mathbb{R}_+$). There remain only two parameters to tune: the partition term coefficient λ, and the common value of the contour regularization terms γ. The parameter λ is first determined with a value large enough in order to ensure at the end of the algorithm that the partition constrainst is satisfied. The results are not sensitive to variations of λ, provided it is large enough. Second, the regularization parameter γ. Variations of γ give more or less regular solutions. This parameter is tuned by trial and error.

Initialization: To get an automatic initialization, and to make it independent of the user, we have then used "seeds": we split the initial image into small sub-images (in practice 5*5 images). In each sub-image, for each class k, we compute the data term by assuming that all the pixels of the sub-image belong to the same class k. We set all the pixels in the sub-image to the class k for which the data term of the whole sub-image is the smallest.

Now that we have introduced this general classification framework, we will focus on the data term in the rest of the paper.

4. Specific data terms

In this section, we first present a data term adapted for the classification of non textured images [33], and we then introduce a one devoted to textured image classification [8].

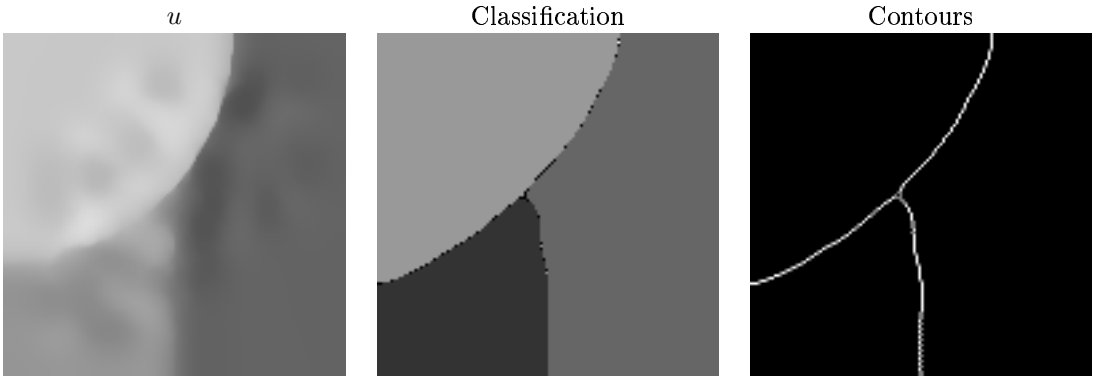

Figure 5: Classification of the geometrical component

Figure 6: Classification of the geometrical component (noisy case)

4.1 Classification of non textured images

In [33], F^C is designed to carry out the classification of non textured images. In that case, the gray level value of each class k is assumed to follow a Gaussian distribution of mean μ_k and standard deviation σ_k. One can then derive the data term F^C by using the maximum likelihood method. We have:

$$B_k(x) = \frac{(u_0(x) - \mu_k)^2}{\sigma_k^2} \tag{4.1}$$

and

$$F^C(\Phi_1, \ldots, \Phi_K) = \sum_{k=1}^{K} e_k \int_\Omega H_\alpha(\Phi_k) \frac{(u_0 - \mu_k)^2}{\sigma_k^2}\, dx \tag{4.2}$$

Numerical examples: We have carried out the classification on the geometrical component u of Figure 2 (resp. Figure 4) . The classification result is displaid on Figure 5 (resp. Figure 6).

The obtained classification result in the noisy case is nearly as good as the one when the original image is noise free.

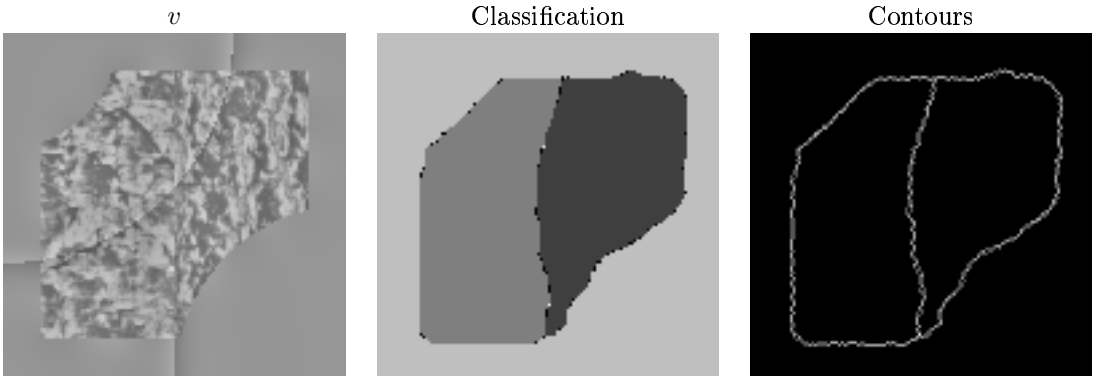

Figure 7: Classification of the textured component

4.2 Classification of textured images

In [8], F^C is designed to carry out the classification of textured images. In that case, a class k is a texture and is characterized through an undecimated wavelet packet transform. To each pixel s, one can associate a vector $U(s) = (U_1(s), \ldots, U_I(s))$, where $U_i(s)$ is the square of the wavelet coefficient in the sub-band i at pixel s. Assuming that U_i has a density probability of the kind (this fact has been checked empirically in [8])

$$p_{U_i}(y) = \frac{A_i}{2\sqrt{y}} \exp\left(-\left(\frac{\sqrt{y}}{\alpha_i}\right)^{\beta_i}\right) \mathbf{1}_{y\geq 0} \tag{4.3}$$

one can derive the data term thanks to the maximum likelihood method:

$$B_k(x) = \sum_{i=1}^{I} \left(-\log A_i^k + \ln 2 + \frac{1}{2}\log u_i(x) + \left(\frac{\sqrt{u_i(x)}}{\alpha_i^k}\right)^{\beta_i^k}\right) \tag{4.4}$$

The parameters A_i, α_i and β_i are computed from the first and second order moments of U_i (which are the parameters given by the user).

Numerical examples: We have carried out the classification on the textured component v of Figure 2 (resp. Figure 4) . The classification result is displaid on Figure 7 (resp. Figure 8).

In the noisy case, the result is not as good as in the case when the original image is noise free, but it is still a good one.

Remark: It is possible to get the segmentation of the original image just by combining the segmentation result we got for the geometrical component with the one we got for the textured component. On Figure 9, we display the final segmentation result we obtain for the image of Figure 1 (by combining the result of Figures 5 and 7), as well as the one we get with the image of Figure 3 (by combining the result of Figures 6 and 8).

These results are interesting, but we want to combine more efficiently the textured and the non textured information. Indeed, on Figure 9, we would like for instance to merge the common boundaries we get from both components. Thanks to the logic framework, we will be able to do it in the next section.

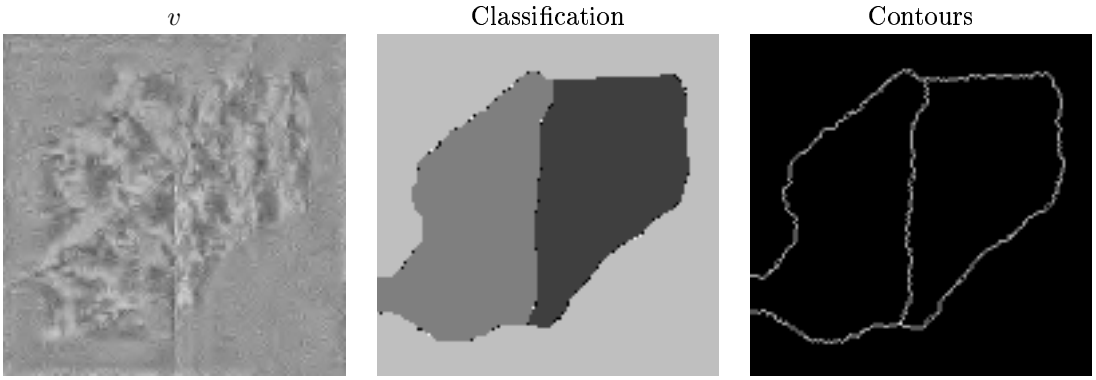

Figure 8: Classification of the textured component (noisy case)

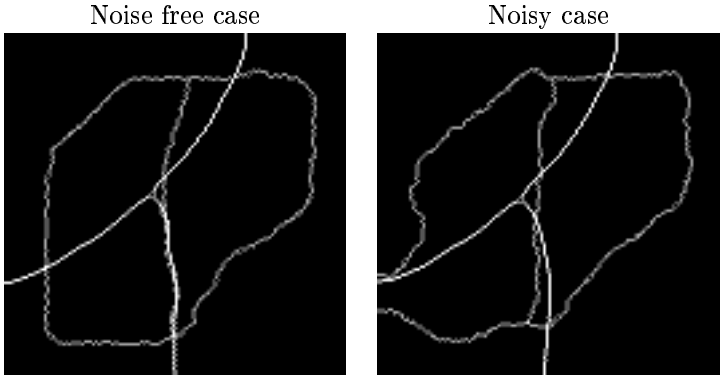

Figure 9: Final segmentation

5. Image classification combining textured and non textured information

In the preceding section, we have presented two different data terms. We now aim at combining them to carry out image classification out of both textured and non textured features. A natural way to achieve this goal is to use the logic framework introduced in [36].

5.1 Logical model

In [36], the authors propose a framework for object detection using logic operations as a structure for defining multi-channel segmentation. The model combines object information from the different channels into any logic combination. In this paper, we consider the logical OR and the logical AND models, although this method extend to the other logic models as well. A comprehensive description of various logic models can be found in [36]. The basic idea is that when we have available several channels corresponding to the same original image, each one contain some interesting information. The difficulty is then to know how to combine all the information coming from the different channels. The simplest idea might be to form an energy by adding the different pieces of information coming from each channels: this is the underlying idea in [36, 8, 31] for instance. But restricting the way to combine the information to this single possibility is obviously too limited. That is why the authors of [36] have developped a logic framework in which any logical operation between the different channels can be performed. This new way of considering multi-channels data has considerably increased the possible applications of active contours methods in image segmentation and classification. An application of the logical framework to joint segmentation and registration has been proposed in [28], and an application to the "ative contours without edges" model [17] has been developped in [36].

We propose here a way to adapt it to our classification framework. The logical OR and logical AND models are built from the two logical elements:

$$\begin{aligned} &\text{OR element} && \sqrt{\tilde{B}_1(x)\tilde{B}_2(x)} \\ &\text{AND element} && 1-\sqrt{(1-\tilde{B}_1(x))(1-\tilde{B}_2(x))} \end{aligned} \tag{5.1}$$

These logical elements are continuous analogues of the OR and AND operations of binary logic. The arguments $\tilde{B}_1$ and $\tilde{B}_2$ take on any real value between 0 and 1, where a value of 0 corresponds to no error and a value of 1 corresponds to the maximum error. By construction, the logical elements also take on values between 0 and 1.

To apply this logical model to our classification framework, we need to modify slightly our data term: we need to calibrate the different data terms. Indeed, the functions B_k which we use for non textured images is positive, but it can be larger than 1; and there is no reason why the B_k we use for textured images should range in $[0,1]$. We know that the larger $B_k(x)$ is, the less likely x belongs to the class k. Therefore, we just need to linearly modify B_k so that it ranges in $[0,1]$ (and so that $B_k(x)=0$ means that x is very likely to belong to the class k, whereas $B_k(x)=1$ means that x is very unlikely to belong to the class k).

To linearly modify the functions B_k, we compute $\max_k = \sup_{x\in\Omega} B_k(x)$, as well as $\min_k = \inf_{x\in\Omega} B_k(x)$. We set $\tilde{B}_k = \frac{B_k - \min_k}{\max_k - \min_k}$. We then use these new functions $\tilde{B}_k$ in the logical framework (5.1).

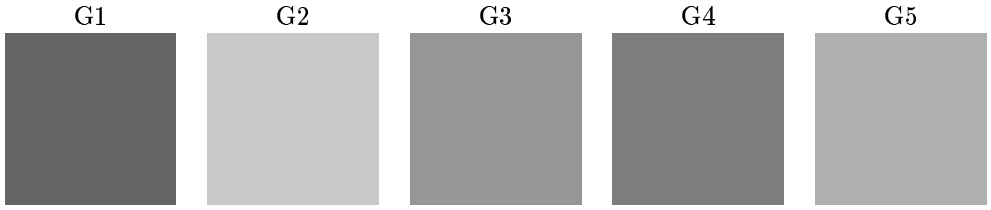

Figure 10: Non-textured patterns

Remark: Another possible choice for the logical expression we use in our classification framework is (for a complete discussion on the choice of the logical expression, we refer the reader to [27]):

$$\begin{array}{rl} \text{OR element} & \min(\tilde{B_1}(x), \tilde{B_2}(x)) \\ \text{AND element} & \max(\tilde{B_1}(x), \tilde{B_2}(x)) \end{array} \tag{5.2}$$

We can use (5.2) instead of (5.1). In practice, it gives similar results. But the choice of (5.2) allows to favour each specific feature of a class. For instance, assume that feature 1 is very likely to appear in the image to classify. Then one can replace $\tilde{B_1}$ in (5.2) by $\mu\tilde{B_1}$ with $\mu \in (0,1)$. If on the contrary, feature 1 is very unlikely to appear in the image, than one can replace $\tilde{B_1}$ in (5.2) by $\mu\tilde{B_1}$ with $\mu > 1$.

At this point of the paper, we have introduced all the tools we need in our classification algorithm. We can now present some numerical results to illustrate our approach in the following section.

5.2 Numerical results

5.2.1 Introduction

We recall here the idea of our algorithm. Given an image, we first split it into two components u and v, u containing the geometrical information of the original image and v the textured information (see Section 2). We then carry out the classification of the image by merging the information from both channels thanks to the logical framework.

We show the different geometrical patterns (resp. textured patterns) we use in this subsection on Figure 10 (resp. Figure 11). In the following examples, the class we seek will be logical expression of these features.

5.2.2 A first example

We display the original image on Figure 12.

In this example, there are 3 classes (see Figures 10 and 11):

1. $Cl_1 = G1 \bigcap T1$
2. $Cl_2 = G2 \bigcup T2$
3. $Cl_3 = (G1 \bigcup G3) \bigcap T3$

The corresponding logic functions are:

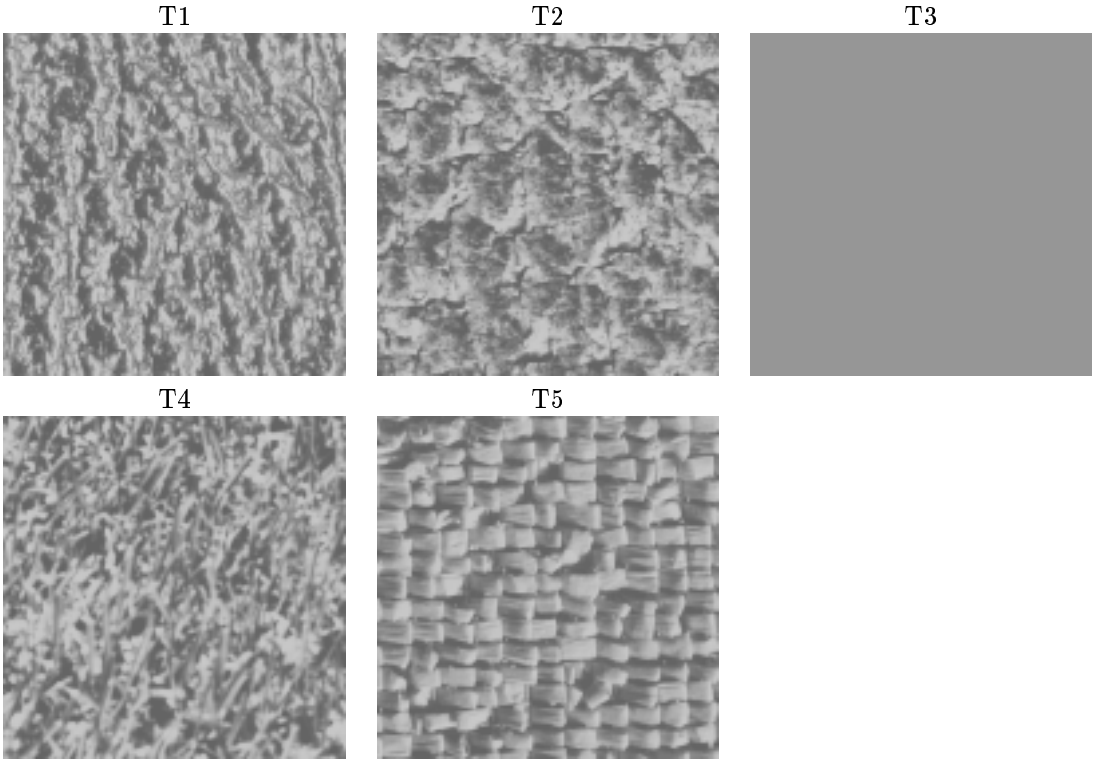

Figure 11: Textured patterns

1. $B_1 = \sqrt{1 - (1 - \tilde{B}_{G_1})(1 - \tilde{B}_{T_1})}$
2. $B_2 = \sqrt{\tilde{B}_{G_2}\tilde{B}_{T_2}}$
3. $B_3 = \sqrt{1 - (1 - \tilde{B}_{G_1}\tilde{B}_{G_3})(1 - \tilde{B}_{T_3})}$

We show the obtained classification result on Figure 13.

Noisy case: We then use our classification algorithm on a noisy version of the image on Figure 14: we corrupt it with some additive Gaussian noise with standard deviation $\sigma = 20$.

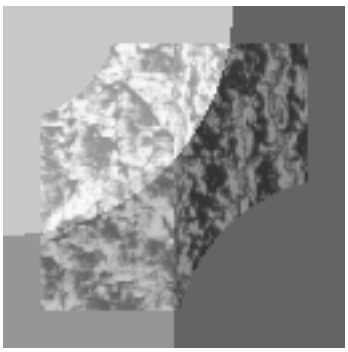

Figure 12: Original synthetic image

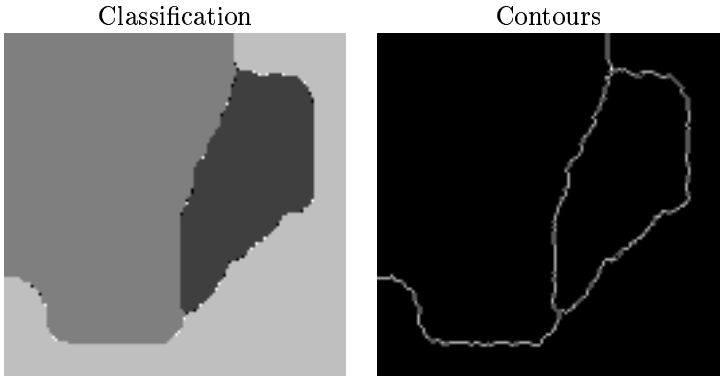

Figure 13: Final classification of Figure 12

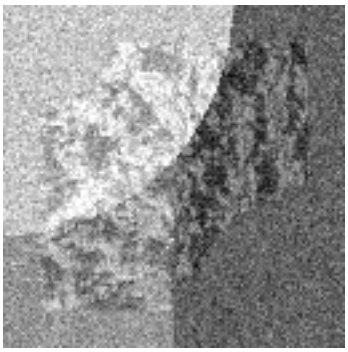

Figure 14: Noisy synthetic image ($\sigma = 20$)

We take the same three classes as before We show the obtained classification result on Figure 15: the result we get is almost as good as the one we had got in the noise free case (see Figure 13).

Using a bin: In theory, we could have replace the charaterization of class Cl_3 in the previous example (noisy or non noisy) by a "no constraint" characterization. This class would have acted as a bin. Any pixel unlikely to belong to Cl_1 or Cl_2 would have been put in Cl_3. To do this, we juste need to set $B_3 = 0.5$ (we assign the middle value to Cl_3). Nevertheless, in that case, the numerical results are not as good as one could expect, and we recommand the use of a "positive" constraint rather than a "no constraint" class.

5.2.3 A second example

We display the original image on Figure 16.

In this example, there are 3 classes (see Figures 10 and 11):

1. $Cl_1 = G2 \bigcap (T3 \bigcup T2)$
2. $Cl_2 = (G2 \bigcup G3) \bigcap (T1 \bigcup T5)$
3. $Cl_3 = G1 \bigcap T2$

The corresponding logic functions are:

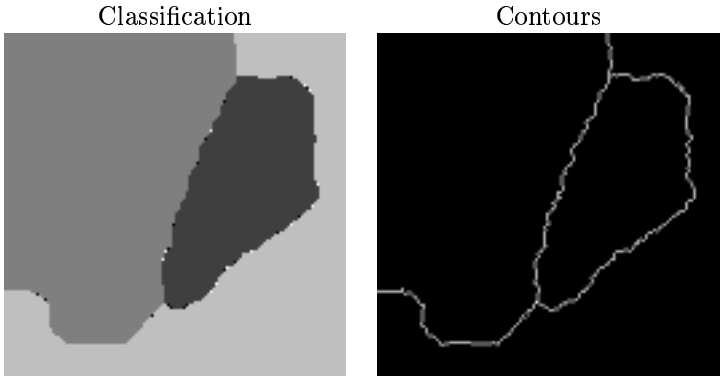

Figure 15: Final classification of Figure 14

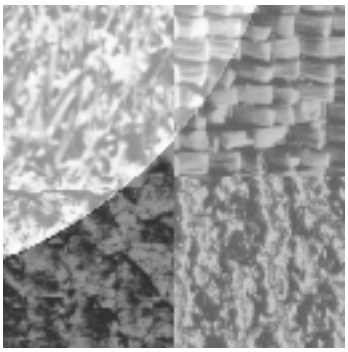

Figure 16: Original synthetic image

1. $B_1 = 1 - \sqrt{(1 - \tilde{B}_{G_2})(1 - \tilde{B}_{T_2}\tilde{B}_{T_3})}$
2. $B_2 = 1 - \sqrt{(1 - \tilde{B}_{G_2}\tilde{B}_{G_3})(1 - \tilde{B}_{T_1}\tilde{B}_{T_5})}$
3. $B_3 = 1 - \sqrt{(1 - \tilde{B}_{G_1})(1 - \tilde{B}_{T_2})}$

We show the decomposition obtained with the $u+v$ algorithm on Figure 17, and the obtained classification result on Figure 18. Even in such a complicated case, our algorithm performs very well.

5.2.4 Limitations of our approach

We display the original image on Figure 19.

In this example, there are 2 classes (see Figures 10 and 11):

1. $Cl_1 = G4 \bigcup T1$
2. $Cl_2 = G5 \bigcap T2$

The corresponding logic functions are:

1. $B_1 = \sqrt{\tilde{B}_{G_4}\tilde{B}_{T_1})}$

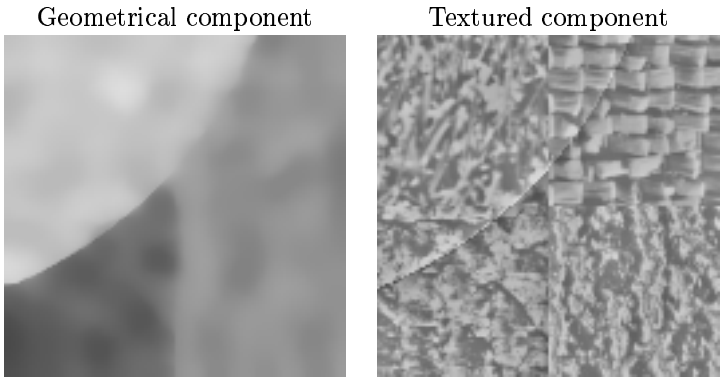

Figure 17: $u + v$ decomposition of Figure 16

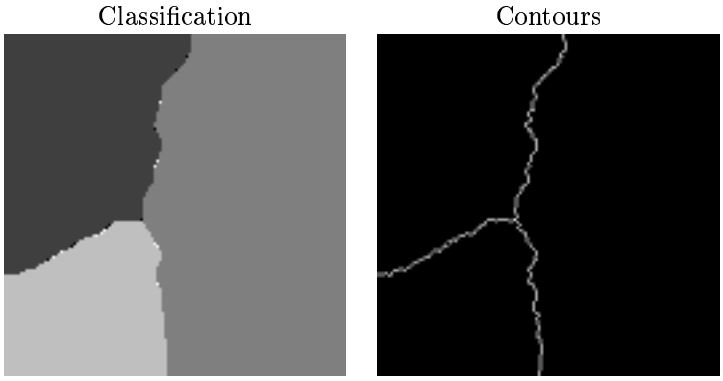

Figure 18: Final classification of Figure 16

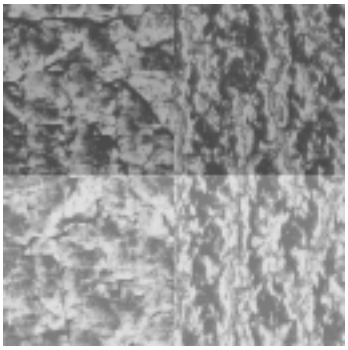

Figure 19: Original synthetic image

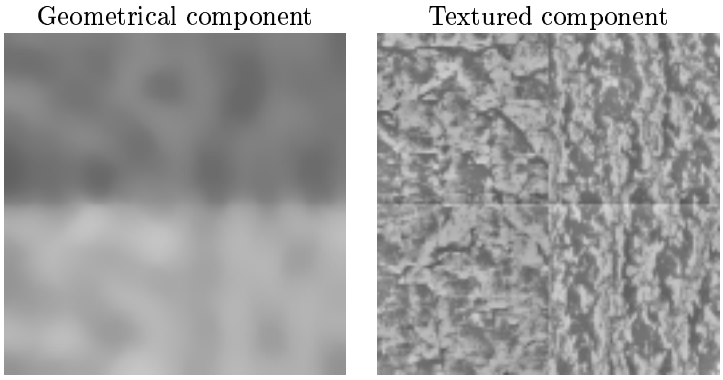

Figure 20: $u + v$ decomposition of Figure 19

2. $B_2 = \sqrt{1 - (1 - \tilde{B}_{G_5})(1 - \tilde{B}_{T_2})}$

We show the decomposition obtained with the $u+v$ algorithm on Figure 20, and the obtained classification result on Figure 21. In this case, we need to enforce the regularization parameter (i.e. the common value $\gamma = \gamma_1 = \cdots = \gamma_K$ of coefficients of the contour lenght penalization term) to get rid of small artifacts. This is the reason why the square representing the class Cl_2 is eroded on its top right. This example clearly illustrates the limitations of our approach. In this case, the two texture classes are visually very close after the decomposition step. And the information coming from the non textured component cannot be of great help.

5.2.5 Finding a car

In this example, we illustrate the capacity of our algorithm at finding objects characterized by both textured and non-textured features. Here, the object to find is a car (see Figure 22).

We display the original image on Figure 23 (car 1).

In this example, there are 2 classes (see Figures 10 and 11):

1. $Cl_1 = G5 \bigcup T1$

2. $Cl_2 = G4 \bigcap T2$

Cl_1 is the class of the car, and Cl_2 the class of the background. The corresponding logic functions are:

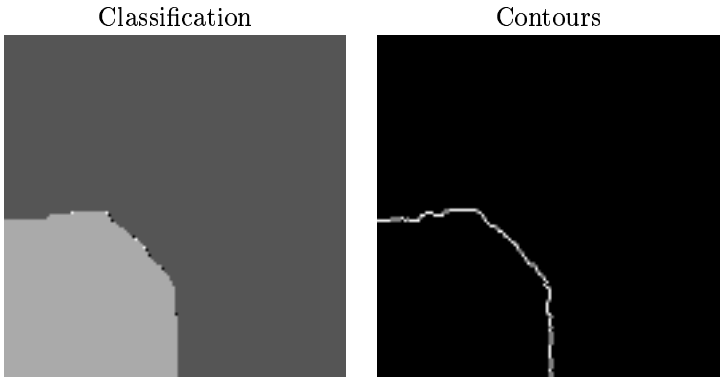

Figure 21: Final classification of Figure 19

Figure 22: Sketch of a car

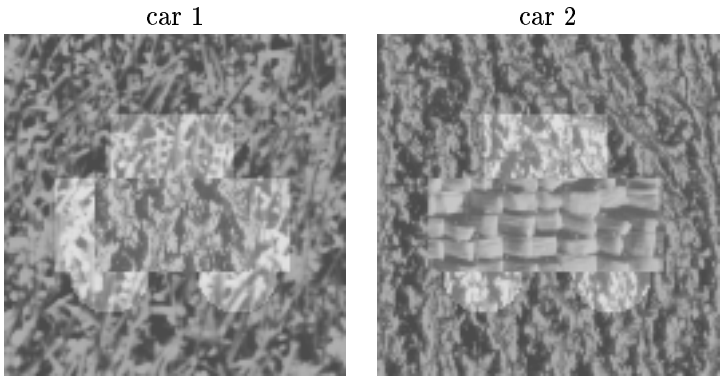

Figure 23: Original synthetic image

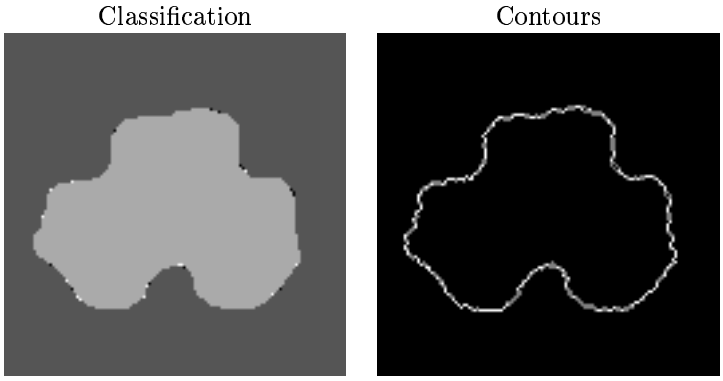

Figure 24: Final classification for car 1 in Figure 23

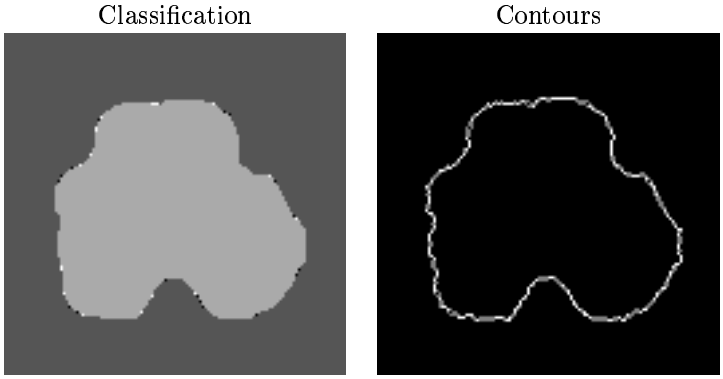

Figure 25: Final classification for car 2 in Figure 23

1. $B_1 = \sqrt{\tilde{B}_{G_5}\tilde{B}_{T_1})}$
2. $B_2 = \sqrt{1-(1-\tilde{B}_{G_4})(1-\tilde{B}_{T_2})}$

We show the obtained classification result on Figure 24.

Finding another car: We display the original image on Figure 23 (car 2).
In this example, there are 2 classes (see Figures 10 and 11):

1. $Cl_1 = G5 \bigcup T5$
2. $Cl_2 = G4 \bigcap T1$

As before, Cl_1 is the class of the car, and Cl_2 the class of the background. The corresponding logic functions are:

1. $B_1 = \sqrt{\tilde{B}_{G_5}\tilde{B}_{T_5})}$
2. $B_2 = \sqrt{1-(1-\tilde{B}_{G_4})(1-\tilde{B}_{T_1})}$

We show the obtained classification result on Figure 25.

Figure 26: Original synthetic image

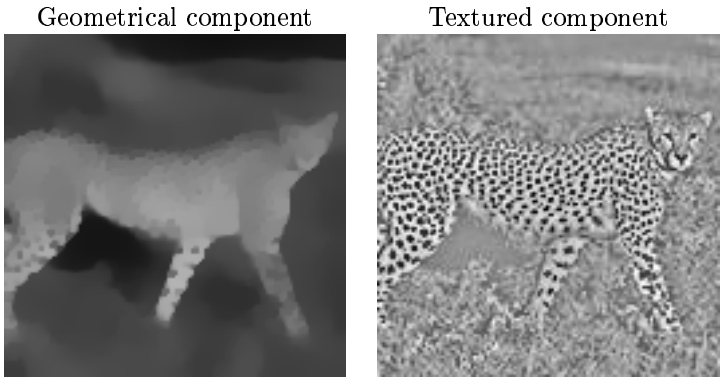

Figure 27: $u + v$ decomposition of Figure 26

Remark: In the case when we aim at finding a given object in the image, we could improve the quality of our segmentation result by using shape priors, as introduced in [18, 19].

5.2.6 Finding a leopard

In this example, we aim at segmenting the leopard in Figure 26. we therefore have two classes. The first one corresponds to the leopard. It is characterized by the intersection of the textured pattern (the dots) of the leopard with the average gray level value of its skin. The second class corresponds to the background of the image. It is characterized by the intersection of the textured pattern (the vegetation) of the landscape with the average gray level value of the background (which is darker than the leopard).

We show the decomposition obtained with the $u+v$ algorithm on Figure 27, and the obtained classification result on Figure 28. Thanks to the combination of both textured and non textured patterns, we get an almost perfect segmentation of the leopard

For comparisons, we have also used the classification algorithms of [33, 8] on the leopard image of Figure 26. We display the result with the algorithm of [33] (resp. [8]) on Figure 29 (resp. Figure 30). The algorithm of [33] is designed for non textured images. It therefore perfoms well on the structures of the image, but fails with the textures: it can not get the legs of the leopard for instance. The algorithm of [8] is suited for texured images. But it fails at recovering the head of the leopard wich is a non textured part of the image. This example clearly illustrates the advantage of our approach. By using both textured and non texured

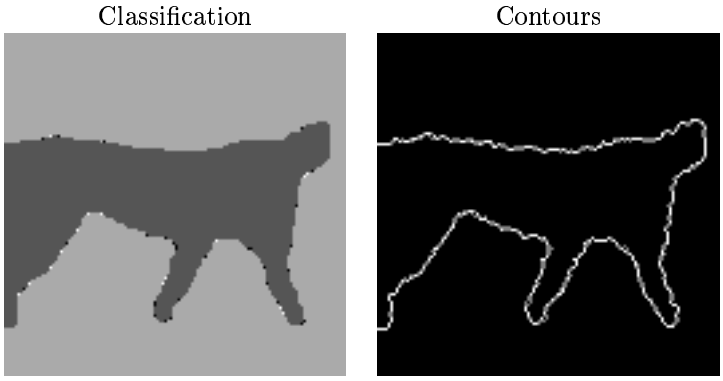

Figure 28: Final classification of Figure 26

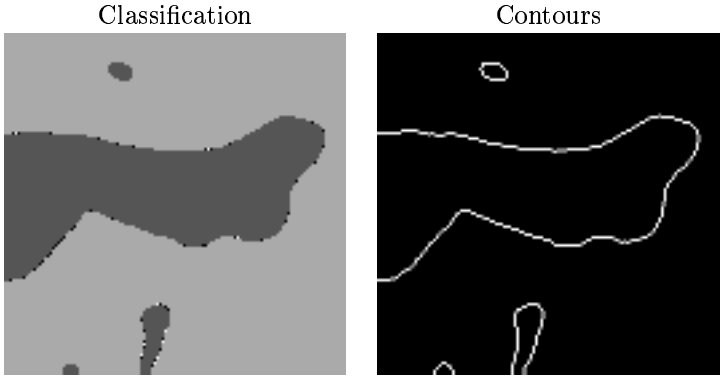

Figure 29: Classification of Figure 26 with the algorithm of [33]

features, our algorithm gives a very good result, wheras the two previous ones fail. In this particular case, the performance of our algorithm also comes from the very nice splitting of the original image into a geometrical component and a textured component (with the algorithm of [7]). The geometrical component is then almost piecewise constant, which is precisely the case when the classification model of [33] is the more efficient. And the textured component has really the same characteristics than in the original image (almost all the textured information has been preserved), so that the classification algorithm of [7] can also provide a good result. The combination of these two information thanks to the logic framework of [36] can therefore lead to this almost perfect segmentation result.

6. Conclusion

In this paper, we have presented a new supervised classification algorithm for images with both textured and non textured areas. This algorithm is based on a variational approach. Thanks to a decomposition algorithm, we split the image into a geometrical part and a textured part [7, 6, 9]. And then, thanks to the logic framework introduced in [36], we can combine the information from both channels to get the classification. In our framework, a class can be characterized both by textured and non-textured features. This generalizes the approaches of [33, 44, 17] which are designed to work with non-textures images, as well as the approaches of [31, 37, 8] which are designed to work with textures images. Since natural images are

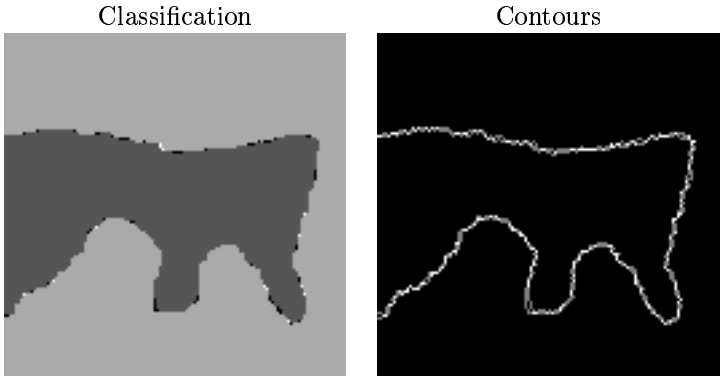

Figure 30: Classification of Figure 26 with the algorithm of [8]

combinations of both textured and non textured patterns, our new algorithm considerably enlarges the scope of possible applications for active contours-based classification algorithms.

In a future work, we intend to extend our approach to the classification of colored images, with still the ability to characterize a class both with textured and non-textured features. We also plan to improve the decomposition part of the algorithm. In particular, we think that relating the G norm to a typical scale in the image would be of great help for tuning the parameters.

References

[1] L. Ambrosio, N. Fusco, and D. Pallara. *Functions of bounded variations and free discontinuity problems*. Oxford mathematical monographs. Oxford University Press, 2000.

[2] G. Aubert and J.F. Aujol. Signed distance functions and viscosity solutions of discontinuous Hamilton-Jacobi equations, July 2002. INRIA Research Report 4507, http://www.inria.fr/rrrt/rr-4507.html.

[3] G. Aubert and J.F. Aujol. Modeling very oscillating signals. Application to image processing, 2003. INRIA Research Report, to appear in Applied Mathematics and Optimization, http://www.inria.fr/rrrt/rr-4878.html.

[4] G. Aubert and J.F. Aujol. Optimal partitions, regularized solutions, and application to image classification, 2004. To appear in Applicable Analysis, http://www.math.ucla.edu/∼aujol/publications.html.

[5] G. Aubert and P. Kornprobst. *Mathematical Problems in Image Processing*, volume 147 of *Applied Mathematical Sciences*. Springer-Verlag, 2002.

[6] J.F. Aujol, G. Aubert, L. Blanc-Féraud, and A. Chambolle. Decomposing an image: Application to SAR images. In *Scale-Space '03*, volume 1682 of *Lecture Notes in Computer Science*, 2003.

[7] J.F. Aujol, G. Aubert, L. Blanc-Féraud, and A. Chambolle. Image decomposition into a bounded variation component and an oscillating component. *JMIV*, 22(1), January 2005.

[8] J.F. Aujol, G. Aubert, and L. Blanc-Féraud. Wavelet-based level set evolution for classification of textured images. *IEEE Transactions on Image Processing*, 12(12):1634–1641, 2003.

[9] J.F. Aujol and A. Chambolle. Dual norms and image decomposition models. *IJCV*, June 2005. in press, http://www.math.ucla.edu/∼aujol/publications.html.

[10] J.F. Aujol and B. Matei. Simultaneous structure and texture compact representation. In *ACIVS 2004*, September 2004.

[11] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher. Simultaneous structure and texture image inpainting. *IEEE Transactions on Image Processing*, 12(8):882–889, 2003.

[12] M. Berthod, Z. Kato, S. Yu, and J. Zerubia. Bayesian image classification using markov random fields. *Image and Vision Computing*, 14(4):285–293, 1996.

[13] C.A. Bouman and M. Shapiro. A multiscale random field model for bayesian image segmentation. *IEEE Trans. on Image Precessing*, 3:162–177, 1994.

[14] V. Caselles, F. Catte, T. Coll, and F. Dibos. A geometric model for active contours. *Numerische Mathematik*, 66:1–31, 1993.

[15] A. Chambolle. An algorithm for total variation minimization and applications. *JMIV*, 20:89–97, 2004.

[16] A. Chambolle, R.A. De Vore, N. Lee, and B.J. Lucier. Nonlinear wavelet image processing: Variational problems, compression, and noise removal through wavelet shrinkage. *IEEE Transcations on Image Processing*, 7(3):319–335, March 1998.

[17] T.F. Chan and L.A. Vese. Active contours without edges. *IEEE Transactions on Image Processing*, 10(2):266–77, February 2001.

[18] D. Cremers and C. Schnörr. Statistical shape knowledge in variational motion segmentation. *Image and Vision Computing*, 27(1):77–86, January 2003.

[19] D. Cremers, F. Tischhäuser, J. Weickert, and C. Schnörr. Diffusion snakes: Introducing statistical shape knowledge into the mumford-shah functional. *IJCV*, 50(3):295–313, December 2002.

[20] I. Daubechies and G. Teschke. Variational image restoration by means of wavelets: simultaneous decomposition, deblurring and denoising, 2004. submitted.

[21] R. Fjortoft, A. Lopes, P. Marthon, and E. Cubero-Castan. An optimal multiedge detector for sar image segmentation. *IEE Transactions on Geoscience and Remote Sensing*, 36(3):793–802, May 1998.

[22] Z. Kato. *Modélisation markoviennes multirésolutions en vision par ordinateur. Application à la segmentation d'images SPOT*. PhD thesis, Université de Nice Sophia Antipolis, 1994. (in French and English).

[23] S. Lakshmanan and H. Derin. Simultaneous parameter estimation and segmentation of gibbs random fields using simulated annealing. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11:799–813, August 1989.

[24] C. Lemarechal, R. Fjortoft, P. Marthon, E. Cubero-Castan, and A. Lopes. Sar image segmentation by morphlogical methods. In *Proc. SAR Image Analysis, Modelling, and Technique III*. SPIE, September 1998.

[25] B. Manjunath and R. Chellappa. Unsupervised texture segmentation using markov random fields models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13:478–482, May 1991.

[26] Yves Meyer. Oscillating patterns in image processing and in some nonlinear evolution equations, March 2001. The Fifteenth Dean Jacquelines B. Lewis Memorial Lectures.

[27] M. Moelich. *Logic models for segmentation and tracking*. PhD thesis, UCLA, 2004. Advisor: T.F. Chan.

[28] M. Moelich and T. F. Chan. Joint segmentation and registration using logic models, 2003. CAM Report 03-06.

[29] S. Osher and R.P. Fedkiw. Level set methods: An overview and some recent results. *Journal of Computational Physics*, 169:463–502, 2001.

[30] S.J. Osher, A. Sole, and L.A. Vese. Image decomposition and restoration using total variation minimization and the H^{-1} norm. *Multiscale Modeling and Simulation: A SIAM Interdisciplinary Journal*, 1(3):349–370, 2003.

[31] N. Paragios and R. Deriche. Geodesic active regions and level set methods for supervised texture segmentation. *International Journal of Computer Vision*, 46(3), 2002.

[32] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.

[33] C. Samson, L. Blanc-Feraud, G. Aubert, and J. Zerubia. A level set method for image classification. *IJCV*, 40(3):187–197, 2000.

[34] C. Samson, L. Blanc-Feraud, G. Aubert, and J. Zerubia. A variational model for image classification and restoration. *IEEE Transactions on Pattern Analyis and Machine Intelligence*, 22(5):460–472, May 2000.

[35] C. Samson, L. Blanc-Féraud, G. Aubert, and J. Zerubia. Two variational models for multispectral image classification. In *Proceedings of the third International workshop on EMMCVPR*, September 2001. Sophia Antipolis, France.

[36] B. Y. Sandberg and T. F. Chan. A logic framework for active contours on multi-channel images, 2002. CAM Report 02-12, to appear in the Journal of Visual Communication and Image Representation.

[37] B. Y. Sandberg, T. F. Chan, and L. A. Vese. A level-set Gabor-based active contours algorithm for segmenting textured images, 2002. CAM Report 02-39.

[38] J.L. Starck, M. Elad, and D.L. Donoho. Image decomposition: separation of texture from piecewise smooth content, 2003. To appear in IEEE Transactions on Image Processing.

[39] M. Sussman, P. Smereka, and S. Osher. A level set approach for computing solutions to incompressible two-phase flow. *Journal of Computational Physics*, 114:146–159, 1994.

[40] G. Unal, A. Yezzi, and H. Krim. Information-theoretic active polygons for unsupervised texture segmentation, June 2002. preprint available upon request, submitted.

[41] M. Unser. Texture classification and segmentation using wavelet frames. *IEEE Transactions on Image Processing*, 4(11):1549–1560, November 1995.

[42] L.A. Vese and T.F. Chan. A multiphase level set framework for image segmentation using the Mumford and Shah model. *International Journal of Computer Vision*, 50(3):271–293, 2002.

[43] L.A. Vese and S.J. Osher. Modeling textures with total variation minimization and oscillating patterns in image processing. *Journal of Scientific Computing*, 19:553–572, 2003.

[44] Hong-Kai Zhao, T. Chan, B. Merriman, and S. Osher. A variational level set approach to multiphase motion. *Journal of Computational Physics*, 127:179–195, July 1996.

[45] Song Chu Zhu and A.Yuille. Region competition: unifying snakes, region growing, and bayes/mdl for multi-band image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9), 1996.