HAL
open science

Image Segmentation with Optimal Control Techniques

Hend Ben Ameur, Guy Chavent, Francois Clement, Pierre Weis

To cite this version:

Hend Ben Ameur, Guy Chavent, Francois Clement, Pierre Weis. Image Segmentation with Optimal Control Techniques. [Research Report] RR-7446, 2010, pp.21. inria-00533799v1

HAL Id: inria-00533799

https://inria.hal.science/inria-00533799v1

Submitted on 8 Nov 2010 (v1), last revised 11 May 2011 (v3)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Image Segmentation with Optimal Control Techniques

Hend Ben Ameur — Guy Chavent — François Clément — Pierre Weis

N° 7446

Novembre 2010

Rapport de recherche

ISSN 0249-6399 ISRN INRIA/RR--7446--FR+ENG

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE
INRIA
centre de recherche PARIS - ROCQUENCOURT

Image Segmentation with Optimal Control Techniques

Hend Ben Ameur*, Guy Chavent† , François Clément† , Pierre Weis†

Thème : Observation et modélisation pour les sciences de l'environnement
Équipe-Projet Estime

Rapport de recherche n° 7446 — Novembre 2010 — 21 pages

Abstract:
We transpose an optimal control technique to the image segmentation problem. The idea is to consider image segmentation as a parameter estimation problem. The parameter to estimate is the color of the pixels of the image. We use the *adaptive parameterization* technique which builds iteratively an optimal representation of the parameter into uniform regions that form a partition of the domain, hence corresponding to a segmentation of the image. We minimize an error function during the iterations, and the partition of the image into regions is optimally driven by the gradient of this error. The resulting segmentation algorithm inherits desirable properties from its optimal control origin: soundness, robustness, and flexibility.

Key-words: image segmentation, adaptive parameterization, inverse problem

* FSB and ENIT-LAMSIN, University of Tunis, BP 37, Le Belvédère, 1002 Tunis, Tunisia. hbenameur@yahoo.ca.
† Projet Estime. {Guy.Chavent,Francois.Clement,Pierre.Weis}@inria.fr.

Centre de recherche INRIA Paris – Rocquencourt
Domaine de Voluceau, Rocquencourt, BP 105, 78153 Le Chesnay Cedex
Téléphone : +33 1 39 63 55 11 — Télécopie : +33 1 39 63 53 30

Segmentation d'image par techniques de contrôle optimal

Résumé :

Nous transposons une technique de contrôle optimal au problème de segmentation d'image. L'idée est de considérer la segmentation d'image comme un problème d'estimation de paramètre où le paramètre à estimer est la couleur des pixels de l'image. Nous utilisons la technique de *paramétrisation adaptative* qui construit itérativement une représentation optimale du paramètre en régions uniformes qui forment une partition du domaine, et correspond ainsi à une segmentation de l'image. Nous minimisons une fonction d'erreur au cours des itérations, et le partitionnement de l'image en régions est piloté de façon optimale par le gradient de cette erreur. L'algorithme de segmentation résultant hérite de propriétés intéressantes de son origine contrôle optimal : fondement, robustesse et flexibilité.

Mots-clés : segmentation d'image, paramtrisation adaptative, problme inverse

1 Introduction

The segmentation of an image is usually defined as the identification of subsets of points with similar image properties, such as the brightness or the color (local properties), or the texture (global property). Image segmentation is usually subdivided into two main classes: edge detection approaches that follow the variations of the image properties, and region-based approaches that follow directly the image properties, e.g. see [19, 27, 26, 2, 15].

Introducing an optimization process to guide a segmentation algorithm has already been investigated. In [14], authors introduce an energy to minimize and develop a "region merging" algorithm for grey level and texture segmentation. The energy method in image segmentation is introduced by Geman and Geman [13]. Later a large variety of energy methods is developed. A formulation of the image segmentation as a piecewise modeling problem combined with an energy method is given in [22]. Uchiyama and Arbib [30] incorporate optimization in a merge-split, region-based image segmentation method; their algorithm converges to a local optimum solution of clustering color space based on the least sum of squares criterion.

The connection between inverse problems and image segmentation has already been noticed, e.g. see [6]. Furthermore, an approach close to the one we are developing can be found in [31], where the segmentation is built through a piecewise least-square approximation of a function related to image intensity. But unlike ours, the refinement of the segmentation is uniform and the refinements are selected from a fixed predefined set.

The *optimal adaptive segmentation algorithm* presented in this paper is also based on an optimization process. At each step, the refinement is optimally obtained from the gradient of the objective function to be minimized. The resulting regions are not geometrically constrained, as they are not required to be connected. Noticeably, all the steps of the algorithm are meaningful, since the algorithm progressively goes from the coarsest segmentation to the finest one, i.e. from a single monochrome region up to the complete reconstruction of the original image with one region per distinct color.

This leads to some kind of *segmentation on demand*: since the nth iteration provides the segmentation into n regions, the algorithm can segment the image into any given number of zones, with each zone having an optimal color with respect to the overall error from the original image. Moreover, since each step of the algorithm selects a single region and splits it into two parts, the adaptive segmentation algorithm can be used to *interactively* refine the segmentation of an image.

This adaptive approach was initially developed for the inverse problem of parameter estimation: given a forward modeling map, the *multidimensional adaptive parameterization* algorithm [4] solves the inverse problem of estimating a multidimensional distributed parameter from measured data. For instance, colors are defined by their three RGB components, thus color images can be seen as three-dimensional distributed parameters. Hence, when the parameter to be identified is a color image and the forward map to be inverted is the identity, this optimal control technique provides naturally an image segmentation algorithm that refines regions guided by the misfit between the original and segmented image.

As illustrated in [4], brute force application of this algorithm to image segmentation gives already interesting results. But the original handling of the multidimensional aspect of the parameter proposed there was rather naive. So this aspect is revisited, developed and enriched in the present paper specifically for the case of image segmentation : beside *vector segmentation* algorithms, where each region is associated with one color, we also introduce *multiscalar segmentation* algorithms, where each color channel has its own segmentation. When transposed back to the general case of nonlinear inverse problem, the new algorithms are also expected to provide more flexibility.

Notice that although adaptive parameterization uses so-called *cuttings* that can be considered as defining edges, this "edge detection" aspect is just an auxiliary step to define the partitioning of the set of pixels into regions. Thus, we still classify our image segmentation algorithm in the region-based techniques.

The paper is organized as follows. In Section 2, we explain the similarity between parameter estimation and image segmentation and we illustrate the usage of adaptive parameterization to segment a simple image. In Section 3, we give some mathematical foundations and the description of the optimal adaptive *vector* and *multiscalar* segmentation algorithms. In Section 4, we use the previous algorithms to segment a color image, and compare their relative performances.

2 From parameter estimation to image segmentation

2.1 Parameter estimation

Parameter estimation is a class of inverse problems, where a simulation model has been constructed but some of its parameters are not precisely known. It consists in trying to recover these parameters from a collection of input-output experimental data.

A classical approach to this inverse problem is to formulate it as the minimization of an objective function defined as a least-squares misfit between experimental output data and the corresponding model-generated quantities:

$$\text{minimize } J(c) = \frac{1}{2}\|d - \mathcal{F}(c)\|^2, \tag{1}$$

where c denotes the unknown parameter, d the data and $\mathcal{F}$ is an operator corresponding to the model describing the cause-to-effect link from c to d.

For a large class of inverse problems, the unknown parameters are space dependent coefficients in a system of partial differential equations modeling a physical problem. Due to the difficulty and high cost of experimental measurements, the available data are often too scarce to allow for the determination of the parameter value in each cell of the computational grid.

Therefore, one has to reduce the number of unknown parameters: a convenient way is to search for unknown distributed parameters as piecewise constant functions, as an approximation to the necessarily more complex but unattainable reality.

Then parameter estimation amounts to identify both the spatial distribution of *regions* of constant value and the associated *parameter values*. The result of the parameter estimation is then a *partition* of the spatial domain into regions together with *one (possibly multi-dimensional) parameter value per region.*

2.2 Image segmentation

Let us now consider the case where the data d is an image, made of a set of pixels (picture elements). Each pixel p_i is the association of a cell x_i with its corresponding color value d_i:

$$d = \{p_i = (x_i, d_i), i \in I\}, \quad \text{where } I \text{ is the set indexing the image pixels.} \tag{2}$$

We shall denote by D the domain of the image, i.e. the set of all its cells x_i:

$$D = \{x_i \ , \ i \in I\}. \tag{3}$$

A *vector segmentation* of d consists in replacing it by an image $c = \{(x_i, c_i), i \in I\}$ (where c stands for "computed") with the same domain such that:

- c has a uniform color $s_{\mathbf{R}}$ (where s stands for "segmented") on each region $\mathbf{R}$ of a partition $\mathcal{P}$ of its domain D:
  $$c_i = s_{\mathbf{R}} \text{ as soon as } x_i \in \mathbf{R} \tag{4}$$
- the image c bears some resemblance to the original image d.

Image *(vector) segmentation* methods are categorized into three main classes: pixel based, edge based and region based approaches [11, 27]. Edge detection is mainly used for gray level image segmentation, it consists in locating points with abrupt changes in gray level, in [16] authors give a survey on techniques for edge detection. Region based approaches include region growing, region splitting, region merging and their combination. They are also divided into two classes: top-down and bottom-up [7]. A combination of region growing and region merging techniques is presented in [29]. In [17, 28], authors use a homogeneous criterion based on a 1D histogram to develop a region splitting technique.

Here we have chosen to ensure the resemblance between the segmented and original images c and d by requiring that their least squares misfit is made "small", or, in more mathematical terms:

$$\text{(5)} \qquad \text{the segmented image } c \text{ minimizes } J(c) = \frac{1}{2}\sum_{i\in I} \|d_i - c_i\|^2$$

under the constraint that the number of regions of $\mathcal{P}$ is "small compared to the number of pixels of d". This approach can be seen as a top-down region based one since we keep refining the regions: at each iteration, the algorithm detects a discontinuity in one region color $s_{\mathbf{R}}$ which give a smaller value to $J(c)$, so the algorithm can also be seen as defining an implicit edge detection.

The result of a *vector segmentation* of an image d is hence a *partition* $\mathcal{P}$ of its domain D into regions, together with the data of *one color value* s_j *per region.* In such a segmentation, all color components use the same partition.

But it is also possible to obtain an image with piecewise constant colors by using a different partition for each color component. We have chosen to use the RGB basis, but any other representation could be used as well. So we define a *multiscalar segmentation* of d as the operation which replaces d by an image $c = \{(x_i, c_i), i \in I\}$ such that:

- for each $k = R, G, B$, the component c^k of c has a uniform intensity $s^k_{\mathbf{R}}$ on each region $\mathbf{R}$ of a partition $\mathcal{P}^k$ of the image domain D:

  $$\text{(6)} \qquad \text{for } k = R, G, B \ \text{ one has: } \ c^k_i = s^k_{\mathbf{R}} \text{ as soon as } x_i \in \mathbf{R},$$

- the image c bears some resemblance to the original image d,

in the sense that c solves the same optimization problem:

$$\text{(7)} \quad \text{the segmented image } c \text{ minimizes } J(c) = \sum_{k=R,G,B} J^k(c) \quad \text{where} \quad J^k(c) = \frac{1}{2}\sum_{i\in I}(d^k_i - c^k_i)^2$$

under the same constraint that the number of regions of the partitions $\mathcal{P}^R, \mathcal{P}^G, \mathcal{P}^B$ is "small compared to the number of pixels of d".

Without the constraint on the number of regions, a trivial exact solution to problem (5) and (7) is the segmentation made of one region for each pixel, with the region color being the color of its unique pixel. However, this trivial exact solution is not satisfactory, since it is not adequate to the segmentation idea. It corresponds to the ultimate level of *over-segmentation.*

The approach proposed in this paper will be able to produce both *vector* and *multiscalar* segmentations, and to control the segmentation level.

2.3 Image segmentation considered as a particular parameter estimation problem

Comparison of the two previous sections shows that vector segmentation of images can be seen as a parameter estimation problem, where the forward model $\mathcal{F}$ reduces to the identity map (compare (1) and (5)) and complete data are available (the exact color is known in each cell).

The parameter estimation problem, because of the nonlinearity of the governing model $\mathcal{F}$ and the scarcity of data, is often ill-posed and difficult to solve. In contrast, one can expect

that the parameter estimation algorithms will work efficiently for vector segmentation, because of the good properties of the associated forward map $\mathcal{F}$ and data. So we adapt in this paper the multidimensional refinement indicators algorithm [4] to the specificities of vector segmentation, and extend it to the determination of multiscalar segmentations.

Remark 1 *The segmentation of an image can also be attempted according to other properties than color, for example texture... The approach developed in this paper can be adapted to these situation by a proper choice of the objective function (5) or (7).*

2.4 Illustration on an example

In the following, we apply the adaptive parameterization algorithm of [4] to perform vector segmentation of simple images.

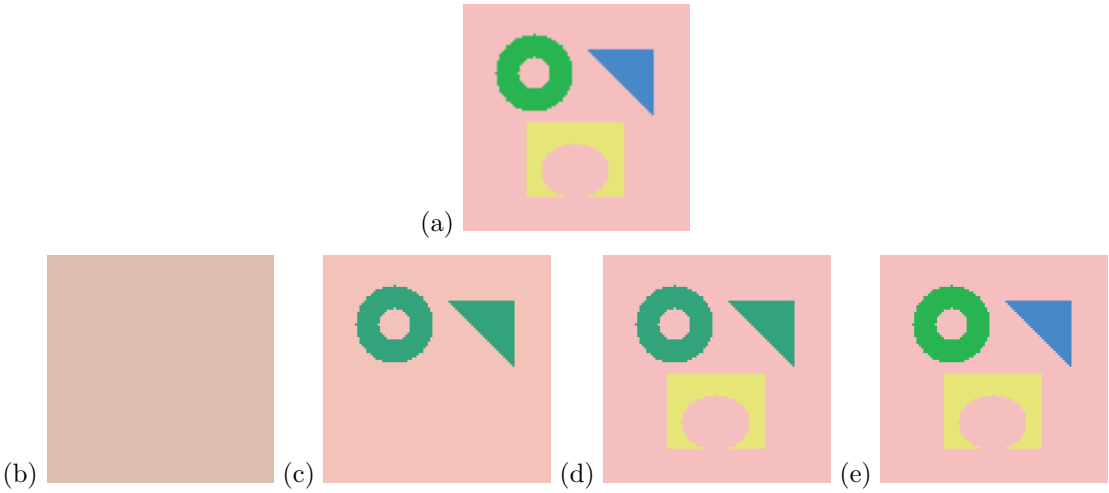

Figure 1: A simple example: (a) is the image to segment, and (b), (c), (d), and (e) are the four successive iterations of the adaptive parameterization algorithm.

In the first example, Figure 1(a) is the original image d, i.e. the *data* of the parameter identification problem. It is made of three geometrical patterns with color green, yellow and blue on top of a pink background. During the iterations illustrated from Figure 1(b) to Figure 1(e), we build the segmentation in an iterative way. We start with one (homogeneous) region in Figure 1(b), it is split into two regions in Figure 1(c), a new region is detected in Figure 1(d), and in Figure 1(e) the algorithm has converged to a segmentation made of four regions, in each region the vector color is constant.

This test case exemplifies the relevance of all the iterations. All iterations are meaningful: the first step gives a monochrome image with the optimal color (the mean color of the picture elements); the second step gives the best splitting of the image into two zones with the best two colors; and so on.

In the second example, the image to be segmented Figure 2(a) is a perturbation of the previous image Figure 1(a) with small square inclusions in the yellow, blue, and green patterns. Iteration results are illustrated from Figure 2(b) to Figure 2(h). The four first iterations do not detect the perturbations, and the fourth Figure 2(e) is exactly the final segmentation of the non perturbed image (to be correct, only the partition associated with it is the same). The last three iterations successively detect the three perturbations.

This second example shows that the larger the contrast is, the faster it is detected, which means that a stopping criterion could be the number of regions supposed to be the more relevant. Also notice that the data image is retrieved in only seven iterations, which is exactly the number of distinct colors in the perturbed data image.

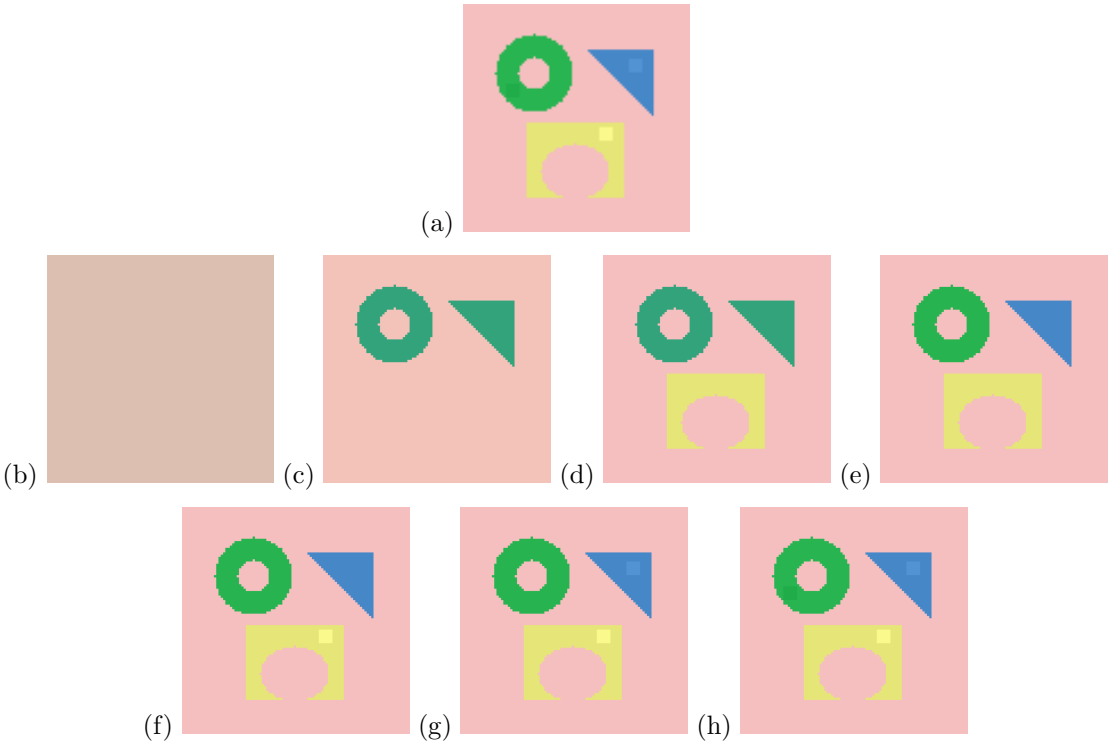

Figure 2: A perturbation of the simple example: (a) is the image to segment, and (b), (c), (d), (e), (f), (g), and (h) are the seven successive iterations of the adaptive parameterization algorithm.

3 Optimal adaptive segmentation

Adaptive segmentation algorithms have been proposed by many authors. In [1] level sets are used to design a robust evolution model based on adaptation of parameters with respect to data. In [25] the adaptivity is related to the fact that the segmentation is adapted to different regions: regions are segmented using an appropriate method according to their types. The adaptivity is achieved through the variation of the size of the neighborhoods used for the estimation of the intensity functions in [18]. Finally, adaptive segmentation in [10] is related to the use of an adaptive clustering algorithm [20] to obtain the specially adaptive dominant colors.

The starting point for the method proposed in this paper is the adaptive multidimensional parameter estimation algorithm proposed in [4] for the general nonlinear inverse problem of section 2.1. It proceeds iteratively, using multidimensional refinement indicators to split at each step one of the regions where the multidimensional parameter is constant. This approach allows to control the issue of *over-parameterization* by stopping the refinement process when the objective function is of the magnitude of the uncertainty or noise level.

As indicated in section 2.3, we shall adapt this algorithm to the specificities of image segmentation. The result appears as an iterative region-based *vector segmentation* method, initialized by a one-region segmented image of uniform color. At each iteration, it adapts the segmentation by splitting one region. The region to be split and the location of the splitting, taken from a set of user-defined admissible cuttings, are chosen according to values of *exact* or *first order refinement indicators* as defined in [9, 4]. These indicators are computed from the objective function (5) or (7) or its gradients with respect to color components. Once the regions are given, the *color* in each region is computed by a straightforward minimization process. With this choice, the new segmented image is likely to produce a significant decrease of the misfit function. The algorithm is stopped when the desired number of regions is attained, i.e. when the segmentation is fine enough

to retain the information of interest, and coarse enough to avoid over-segmentation. Note that regions need not to be necessarily connected in our approach.

We describe in the next sections the specification of the adaptive parameterization technique to the *vector segmentation* problems (5), and its extension to the *multiscalar segmentation* problem (7). We recall that the original image is denoted by $d = \{(x_i, d_i), i \in I\}$ (data), the segmented image by $c = \{(x_i, c_i), i \in I\}$ (computed).

3.1 Case of vector segmentation.

Let $\mathcal{P}$ be the current partition. The associated segmented image c, solution of the least square problem (5) for the partition $\mathcal{P}$, is given by $c = \mathcal{M}_{\mathcal{P}}\tilde{\mathcal{M}}_{\mathcal{P}}\, d$, where $\mathcal{M}_{\mathcal{P}} : \{s_{\mathbf{R}}, \mathbf{R} \in \mathcal{P}\} \mapsto c$ defined by (4) denotes the mapping that paints the pixels of each region $\mathbf{R}$ in $\mathcal{P}$ with a given color $s_{\mathbf{R}}$, and $\tilde{\mathcal{M}}_{\mathcal{P}} = (\mathcal{M}_{\mathcal{P}}^T\mathcal{M}_{\mathcal{P}})^{-1}\mathcal{M}_{\mathcal{P}}^T$ is its least-squares pseudoinverse, which computes the mean value in each region. Hence once the partition $\mathcal{P}$ has been chosen, the best segmented image c is simply obtained by replacing on each region $\mathbf{R}$ the image color by its mean value.

So we concentrate now on the determination of a partition $\mathcal{P}$ with a controlled number of regions and such that $J(c)$ is as small as possible.

3.1.1 Refinement indicators: how good is a cutting?

Let us first denote by C the cutting which splits a region $\mathbf{R}$ of $\mathcal{P}$ into two subregions (C_+, C_-). In order to evaluate the effect of this additional degree of freedom on the decrease of the objective function $J(c)$ in the vector segmentation problem (5), one can use refinement indicators ([9] pp. 110–111):

Exact indicators. They consist in

$$\Delta J = J(c) - J(c_C) \tag{8}$$

where c_C is the solution of problem (5) for the partition $\mathcal{P}_C$ obtained by splitting region $\mathbf{R}$ of $\mathcal{P}$ into C_+ and C_-.

These indicators are called "nonlinear" in [9] because in the general case where $\mathcal{F}$ is nonlinear, the evaluation of c_C requires the resolution of the full nonlinear least squares problem (1) with the tentative refined partition $\mathcal{P}_C$. This prohibits their use for the test of a large number of cuttings.

On the contrary, for our problem where $\mathcal{F}$ reduces to the identity map, J is additive with respect to regions of $\mathcal{P}$, so that c_C *coincides with* c *outside of* $\mathbf{R}$. Its value on $\mathbf{R}$ is given by the simple formulas:

$$\left(c_C\right)_i = \begin{cases} c_+ & \text{for } x_i \in C_+ \quad \text{where } c_+ = \text{mean value of } \{d_j, x_j \in C_+\}\ , \\ c_- & \text{for } x_i \in C_- \quad \text{where } c_- = \text{mean value of } \{d_j, x_j \in C_-\}. \end{cases} \tag{9}$$

Hence ΔJ can be computed at a very low cost. This makes it possible to use exact indicators to rank a quite large number of tentative cuttings.

First order indicators, as defined in [4] for multidimensional parameters, are given by :

$$\|\lambda\|_q, \quad 1 \le q \le \infty, \quad \lambda = (\lambda^R, \lambda^G, \lambda^B) \tag{10}$$

where the q is user defined, and where:

$$\lambda^k = \sum_{x_i \in \mathbf{C}_+} \frac{\partial J^k}{\partial c_i^k} - \sum_{x_i \in \mathbf{C}_-} \frac{\partial J^k}{\partial c_i^k} \quad \text{for } k = R, G, B. \tag{11}$$

Because of the simple form of J^k, the following relation holds for each color component (see [4], or [9] page 126 for the proof):

$$\Delta J^k = \frac{(\lambda^k)^2}{8}\,\frac{p}{p_+^k p_-^k} \quad \text{for } k = R, G, B, \tag{12}$$

where p is the number of points x_i in $\mathbf{R}$ and p_+^k, p_-^k are the numbers of points x_i in the tentative subregions C_+ and C_-.

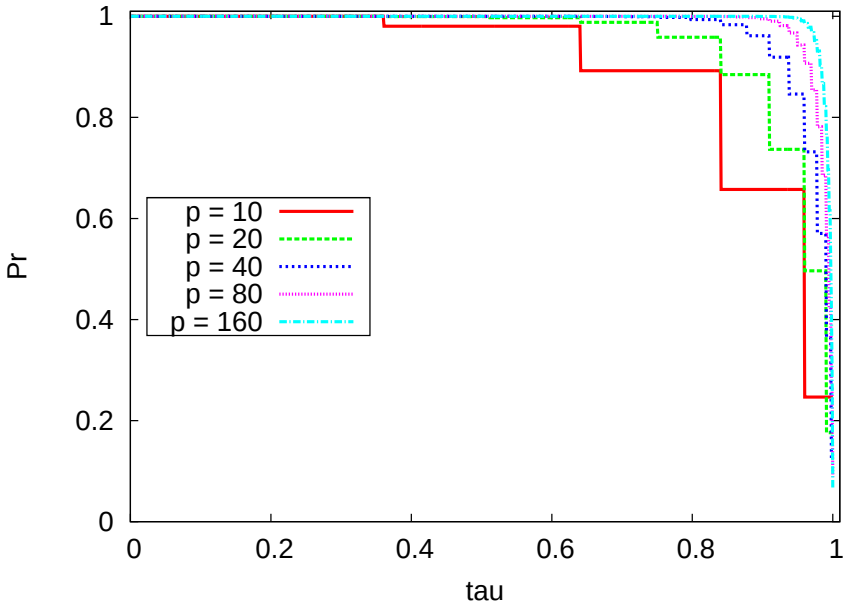

Figure 3: Comparison of exact and first order indicators for $p = 10$ (red), 20 (green), 40 (blue), 80 (pink) and 160 (light blue). Each curve corresponds to the $\tau \mapsto Pr$ function defined in (13).

In order to illustrate the quality of this first order indicator, we have plotted in Figure 3.1.1 the curves:

$$\tau \in [0,1] \mapsto Pr\left\{\tau \leq \frac{(\lambda^k)^2}{2p} \Big/ \Delta J^k \leq 1\right\}. \tag{13}$$

for regions $\mathbf{R}$ with different numbers p of pixel, under the hypothesis that all partitions of $\mathbf{R}$ are equiprobable. As one can see, this probability is practically one for $\tau = 0.9$ as soon as $\mathbf{R}$ has more than 80 pixels! This shows that the first order indicators $|\lambda^k|$ are well suited to rank cuttings inside a given region $\mathbf{R}$. It also shows that they are not suited to rank cuttings belonging to different regions, unless they are replaced by $|\lambda^k|/p^{1/2}$, where p is the number of pixel of the region subject to the cutting.

The computational cost of first order indicator is here similar to that of exact indicators, so one can wonder why to use an approximation when an exact calculation is available at the same price! The answer will appear in the next section.

3.1.2 Cutting strategies

Now that we know how to rank cuttings, the next step is to specify the *cutting strategy*, i.e. the family T of cuttings C to be tested for the splitting of each region.

For nonlinear problems, adaptive parameterization has been shown to be regularizing when the family T is adapted to the problem [5]. But if enriching the cutting family always provides a better local decrease of the cost function, it can also decrease the regularizing effect by leading the algorithm into a local minimum.

However, for our segmentation problem, the forward map $\mathcal{F}$ is the identity, so there is no danger of local minima, and we are free to choose the family T at our convenience, even if it is extremely large. So we shall consider two *cutting strategies*:

Best in a family. The set T of cuttings used for the tentative splitting of a region is a *user-defined family* of cuttings. It may incorporate any a priori information on the parameter when such information is available.

We have used in our numerical results the set T made of all *vertical* and *horizontal* cuttings, which split any region $\mathbf{R}$ in two subregions separated by one vertical or horizontal boundary. This family of cuttings has proven useful for the nonlinear parameter estimation problem, where one seeks to describe the parameter by regions with simple forms (see for example [3], where T contained also diagonal and checkerboard cuttings). But we shall see in the numerical results section that it seems much less adapted to image segmentation.

In any case, the number of cuttings to be tested with this strategy is bounded from above by the sum of the number of horizontal and vertical pixels in the image to be segmented. Hence the best tentative cutting $C^\star_{\mathbf{R}}$ of region $\mathbf{R}$ can be determined by an exhaustive computation of all *exact indicators* ΔJ associated with all cuttings C of $\mathbf{R}$ in the chosen family T, as they are not more expensive as the first order ones:

$$C^\star_{\mathbf{R}} \text{ maximizes } \Delta J \text{ over all cuttings } C \in T \text{ of } \mathbf{R}. \tag{14}$$

Overall best. Here T is made of *all possible 2-partitions* of the region $\mathbf{R}$. So the number of cuttings to be tested for the determination of $C^\star_{\mathbf{R}}$ is $2^{\mathrm{card}(\mathbf{R})}$, which becomes extremely large as soon as $\mathbf{R}$ contains enough points. Hence one cannot afford to perform an exhaustive computation of all corresponding indicators, let them be exact or first order! So this cutting strategy can be used only in cases where an analytical determination of the cut $C^\star_{\mathbf{R}}$ which maximizes ΔJ or $\|\lambda\|_q$ over T is available. To our best knowledge, such analytical solution is available neither for the exact indicator ΔJ, nor for the refinement indicators $\|\lambda\|_q$ for any q ... except for $q = \infty$! This is where the first order indicators play a crucial role in image segmentation.

We describe now the analytical determination of $C^\star_{\mathbf{R}}$ when $q = \infty$. As it was noticed in [4], formula (11) shows that, for each color component k, the cutting $C^{\star k}_{\mathbf{R}}$ which maximizes $|\lambda^k_{\mathbf{R}}|$ is obtained by splitting $\mathbf{R}$ according to the sign of $\partial J^k/\partial c^k_i$:

$$C^{\star k}_{\mathbf{R},+} = \{x_i \in \mathbf{R} \mid \partial J^k/\partial c^k_i \geq 0\}, \quad C^{\star k}_{\mathbf{R},-} = \{x_i \in \mathbf{R} \mid \partial J^k/\partial c^k_i < 0\}. \tag{15}$$

The associated first order indicators of largest modulus for the splitting of $\mathbf{R}$ are then, for each component:

$$\lambda^{\star\, k}_{\mathbf{R}} = |\lambda^{\star\, k}_{\mathbf{R}}| = \sum_{x_i \in \mathbf{R}} \left| \frac{\partial J^k}{\partial c^k_i} \right| \quad \text{for } k = R, G, B. \tag{16}$$

So if we denote by $k^\star$ the color component with the largest $|\lambda^{\star k}_{\mathbf{R}}|$, one has:

$$\begin{cases} \|\lambda\|_\infty \overset{\text{def}}{=} \max\limits_{k=R,G,B} |\lambda^k_{\mathbf{R}}| \leq \max\limits_{k=R,G,B} |\lambda^{\star\, k}_{\mathbf{R}}| = |\lambda^{\star\, k\star}_{\mathbf{R}}|, \\ \|\lambda\|_\infty = |\lambda^{\star\, k\star}_{\mathbf{R}}| \quad \text{for the cutting } \; C = C^{\star\, k\star}_{\mathbf{R}}. \end{cases} \tag{17}$$

This shows that $C^\star_{\mathbf{R}} \overset{\text{def}}{=} C^{\star\, k\star}_{\mathbf{R}}$ maximizes $\|\lambda\|_\infty$ over the set T of all 2-partitions of region $\mathbf{R}$.

Since we have no theoretically funded way to choose a predefined set of cuttings among which to select the refinement, the *overall best* cutting strategy will be the default option for image segmentation.

3.1.3 Updating the partition $\mathcal{P}$

The algorithm computes the optimal cuttings $C^\star_{\mathbf{R}}$ for each region $\mathbf{R}$ of the current partition $\mathcal{P}$ as indicated in the previous section 3.1.2, and determines the region $\mathbf{R}^\star$ with the largest *exact* refinement indicator.

The next partition $\mathcal{P}^\star$ is then obtained by splitting region $\mathbf{R}^\star$ according to cutting $C^\star_{\mathbf{R}^\star}$, thus increasing the number of regions by one. The new segmented image $c^\star$ is updated by formula (9) over the region $\mathbf{R}^\star$, and coincides with c over all other regions.

After n iterations, the *vector segmentation algorithm* adds n regions to the initial partition.

3.1.4 Optimal adaptive *vector segmentation* algorithm

We summarize here the algorithm for the vector segmentation of an image d. Let $I(C)$ denote the *best available* refinement indicators for the ranking of cuttings C in the chosen family T (see section 3.1.2).

Specify the cutting strategy. Choose one from:

- *best in a family* (and then $I(C) = \Delta J =$ exact indicators),
- *overall best* (and then $I(C) = \|\lambda\|_\infty =$ first order indicators).

Initialization.

-1. Define the initial partition $\mathcal{P}_1$ as the 1-region partition of the domain D of d.
0. Compute the associated 1-region segmented image:

$$c_1 = \mathcal{M}_{\mathcal{P}_1} \tilde{\mathcal{M}}_{\mathcal{P}_1} d \quad \Longleftrightarrow \quad \forall i \in I, (c_1)_i = \text{ mean value of } \{d_j, j \in I\}.$$

Iterations. For $n \geq 1$, do until the desired number of regions is attained:

1. For all regions $\mathbf{R}$ of the current partition $\mathcal{P}_n$,
   (a) determine the best cutting $C^\star_{\mathbf{R}}$ using the *best available* indicator $I(C)$,
   (b) compute the *exact indicators* ΔJ associated with $C^\star_{\mathbf{R}}$ using (8) and (9).
2. Retain the region $\mathbf{R}^\star$ whose best cutting $C^\star_{\mathbf{R}^\star}$ has the largest ΔJ.
3. Define the next partition $\mathcal{P}_{n+1}$ by splitting $\mathbf{R}^\star$ into $C^\star_{\mathbf{R}^\star,+}$ and $C^\star_{\mathbf{R}^\star,-}$.
4. Define the next segmented image $c_{n+1} = \mathcal{M}_{\mathcal{P}_{n+1}} \tilde{\mathcal{M}}_{\mathcal{P}_{n+1}} d$ by updating c_n on $\mathbf{R}^\star$ according to the tentative segmented image on $\mathbf{R}^\star$ obtained during step 1(b).

Remark 2 *When $\mathcal{F}$ is not the identity operator, one can use, for the* best in a family *cutting strategy, the first order indicators in step 1(a) instead of the exact ones if the cost of an exhaustive search with the exact indicators is too high (see the adaptive parameterization algorithm [4]).*

Remark 3 *It is of course possible to start the algorithm from any partition $\mathcal{P}_1$ instead of a* 1-*partition ...*

3.2 Case of multiscalar segmentation

In order to construct a multiscalar segmentation, one notice that (7) is equivalent to

$$\text{minimize } J^k(c^k) = \frac{1}{2}\sum_{i\in I}(d_i^k - c_i^k)^2 \quad \text{for } k = R, G, B, \tag{18}$$

under the same constraint that the number of regions of the partitions $\mathcal{P}^k$ is "small compared to the number of pixels of d".

This suggests to apply a scalar version of the previous optimal adaptive segmentation algorithm *independently to each component* $k = R, G, B$. If $\mathcal{P} = (\mathcal{P}^R, \mathcal{P}^G, \mathcal{P}^B)$ denotes the current multiscalar partition, this will produce three *tentative optimal partitions* $(\mathcal{P}^{R\#}, \mathcal{P}^{G\#}, \mathcal{P}^{B\#})$. It will then be necessary to choose one *multiscalar strategy* to decide how to use this information to define the updated multiscalar segmentation $\mathcal{P}^\star = (\mathcal{P}^{R\star}, \mathcal{P}^{G\star}, \mathcal{P}^{B\star})$.

3.2.1 Scalar optimal adaptive segmentation

For a given color component k, the *scalar algorithm* follows the steps of section 3.1.4 with the following minor adaptations.

Let $\mathcal{P}^k$ be the current partition for component k. The associated k-th component segmented image, solution of (18) for the partition $\mathcal{P}^k$, is now given by $c^k = \mathcal{M}^k_{\mathcal{P}^k}\tilde{\mathcal{M}}^k_{\mathcal{P}^k}d^k$, where $\mathcal{M}^k_{\mathcal{P}^k} : (s^k_{\mathbf{R}}, \mathbf{R} \in \mathcal{P}^k) \mapsto c^k$ denotes the mapping that attributes the segmented intensity $s^k_{\mathbf{R}}$ to color component k on region $\mathbf{R}$ of partition $\mathcal{P}^k$, and $\tilde{\mathcal{M}}^k_{\mathcal{P}^k} = \big((\mathcal{M}^k_{\mathcal{P}^k})^T\mathcal{M}^k_{\mathcal{P}^k})^{-1}(\mathcal{M}^k_{\mathcal{P}^k})^T$ its least-squares pseudoinverse, which computes the mean value of the k-th component over each region.

The refinement indicators associated with a cutting C which splits a region $\mathbf{R}$ of the current k-th component partition $\mathcal{P}^k$ into two subregions (C_+, C_-) are (compare with section 3.1.1):

Exact indicators:

(19) $$\Delta J^k = J^k(c^k) - J^k(c^k_C),$$

where c^k is the k-th color component of the current segmented image c, and c^k_C minimizes J^k defined in (18) for the partition $\mathcal{P}^k_C$ obtained by splitting region $\mathbf{R}$ of $\mathcal{P}^k$ into C_+ and C_-. As in the vector case, c^k_C is easily computed on $\mathbf{R}$ by:

(20) $$\left(c^k_C\right)_i = \begin{cases} c^k_+ & \text{for } x_i \in C_+ \quad \text{where } c^k_+ = \text{ mean value of } \{d^k_j, x_j \in C_+\}, \\ c^k_- & \text{for } x_i \in C_- \quad \text{where } c^k_- = \text{ mean value of } \{d^k_j, x_j \in C_-\}. \end{cases}$$

First order indicators:

(21) $$|\lambda^k| \text{ with } \lambda^k \text{ given by (11).}$$

Of course, exact and first order indicators are still linked by relation (12).

In the absence of specific information for each component, we have chosen to use *the same set* T of tentative cuttings for each color component k. So the cutting strategies are (compare to section 3.1.2):

Best in a family. T is a *predefined family* of cuttings, for example the set of vertical and horizontal cuttings. With this strategy, the best cutting $C^\star_{\mathbf{R}}$ of a region $\mathbf{R}$ of $\mathcal{P}^k$ for the *exact indicator* ΔJ^k needs to be computed by an exhaustive search.

Overall best. T is made of *all possible 2-partitions* of the region $\mathbf{R}$. Here the best cutting $C^\star_{\mathbf{R}}$ for the *first order indicators* $|\lambda^k|$ can be determined analytically by formula (16).

So after one iteration of the scalar algorithm has been performed on each component k, the following quantities are available:

(22) $$\mathcal{P}^{k\,\#} = \text{ tentative optimal partition for color component } k,$$

(23) $$\Delta J^{k\,\#} = \text{ exact decrease of the minimum of } J^k \text{ when } \mathcal{P}^k \text{ is replaced by } \mathcal{P}^{k\,\#},$$

3.2.2 Updating the partition $\mathcal{P} = (\mathcal{P}^R, \mathcal{P}^G, \mathcal{P}^B)$: multiscalar strategy

We describe now three *multiscalar strategies* which can be applied to update the current multiscalar partition $\mathcal{P}$ once the results (22), (23) of the scalar algorithm have been obtained for each color component:

Best component only. This strategy determines the color component $k^\star$ whose tentative refinement produces the largest decrease $\Delta J^{k\,\#}$, and applies the corresponding optimal partition $\mathcal{P}^{k^\star\,\#}$ to component $k^\star$ only. When $k^\star = R$ (red) for example, the current and updated multiscalar partitions are:

(24) $$\mathcal{P} = (\mathcal{P}^R, \mathcal{P}^G, \mathcal{P}^B), \quad \mathcal{P}^\star = (\mathcal{P}^{R\,\#}, \mathcal{P}^G, \mathcal{P}^B).$$

After n iterations, the *best component only* multiscalar strategy adds n regions to the initial multiscalar partition. Of course, the vector partition corresponding to the resulting multiscalar partition, obtained by superimposing the three scalar partitions, may contain many more regions.

Best component for each. In this strategy, the algorithm refines each component k according to the corresponding tentative optimal partition $\mathcal{P}^{k\,\#}$ determined by the scalar algorithms. The current and updated multiscalar partitions are then:

$$\mathcal{P} = (\mathcal{P}^R, \mathcal{P}^G, \mathcal{P}^B), \quad \mathcal{P}^\star = (\mathcal{P}^{R\,\#}, \mathcal{P}^{G\,\#}, \mathcal{P}^{B\,\#}). \tag{25}$$

After n iterations, the *best component for each* multiscalar strategy has added $3n$ regions to the initial partition (n regions per component). Again, the vector partition corresponding to the resulting multiscalar partition, obtained by superimposing the three scalar partitions, may contain many more regions.

The *best component for each* multiscalar strategy amounts to apply the scalar refinement indicator algorithm to each component separately, i.e. to use the *tensor product* of the three algorithms.

Combine best components. Here the algorithm selects the three optimal cuttings $C^{k\,\star}_{\mathbf{R}^\star}$ corresponding to the best tentative segmentation for each component $k = R, G, B$, and applies *all of them* to *all components*. This makes sense only if the three current component partitions coincide, and produces a partition $\mathcal{P}^{RGB\,\#}$, which is nothing but the superimposition of the three tentative partitions $\mathcal{P}^{R\,\#}$, $\mathcal{P}^{G\,\#}$ and $\mathcal{P}^{B\,\#}$. This partition is used to update all three components:

$$\mathcal{P} = (\mathcal{P}^R, \mathcal{P}^G, \mathcal{P}^B), \quad \mathcal{P}^\star = (\mathcal{P}^{RGB\,\#}, \mathcal{P}^{RGB\,\#}, \mathcal{P}^{RGB\,\#}). \tag{26}$$

So the *combine best components* multiscalar strategy constructs in fact a sequence of *vector segmentations* of the original image d.

In contrast with the vector segmentation algorithm of section 3.1.4, the *combine best components* multiscalar strategy adds at each iteration between 3 and $2^3 - 1$ regions to the current vector partition. So the number of regions may increase rapidly, which is not compatible with the idea of adaptive parameterization.

Note that all multiscalar strategies are equivalent in the scalar case.

3.2.3 The multiscalar adaptive segmentation algorithm

We summarize here the adaptation of the algorithm of section 3.1.4 to the determination of multiscalar segmentations.

Specify the cutting strategy. Choose one from:

- *best in a family* (best available = exact indicators $I^k(C) = \Delta J^k$),
- *overall best* (best available = first order indicators $I^k(C) = |\lambda^k|$).

Specify the multiscalar strategy. Choose one from:

- *best component only*,
- *best component for each*,
- *combine best components*.

Initialization. Define $\mathcal{P}_1 = (\mathcal{P}_1^R, \mathcal{P}_1^G, \mathcal{P}_1^B)$ and $c_1 = (c_1^R, c_1^G, c_1^B)$ by:

Scalar initializations. For each component $k = R, G, B$:

-1. Define the $\mathcal{P}_1^k$ as the 1-region partition of the domain D of d.
0. Compute the corresponding 1-region segmented *component image*:

$$c_1^k = \mathcal{M}^k_{\mathcal{P}_1^k} \tilde{\mathcal{M}}^k_{\mathcal{P}_1^k} d \quad \Longleftrightarrow \quad \forall i \in I, (c_1^k)_i = \text{ mean value of } \{d_j^k, j \in I\}.$$

Iterations For $n \geq 1$, do until the desired number of regions is attained:

Scalar iterations. For each component $k = R, G, B$, do (section 3.2.1):

1. For all regions $\mathbf{R}$ of the current partition $\mathcal{P}_n^k$,
   (a) determine the best cutting $C^\star_{\mathbf{R}}$ using the *best available* indicator $I(C)$,
   (b) compute the *exact indicators* ΔJ^k associated with $C^\star_{\mathbf{R}}$ using (19) and (20).
2. Retain the region $\mathbf{R}^\star$ whose best cutting $C^{k\star}_{\mathbf{R}^\star}$ has the largest *exact* indicator ΔJ^k.
3. Define the next tentative partition $\mathcal{P}^{k\,\#}$ of color component k by splitting the region $\mathbf{R}^\star$ of $\mathcal{P}^k$ into $C^\star_{\mathbf{R}^\star,+}$ and $C^\star_{\mathbf{R}^\star,-}$.

Update the multiscalar segmentation.

4. Compute the next multiscalar partition $\mathcal{P}_{n+1}$ according to the chosen multiscalar strategy (section 3.2.2).
5. Define the next multiscalar segmented image c_{n+1}:
   - *best component only*: let $k^\star$ be the component retained at step 4. It suffices then to update $c_n^{k^\star}$ on the region $\mathbf{R}^\star$ determined at step 2 according to the tentative segmented image (20) determined at step 1(b).
   - *best component for each*: each component k of c_n^k needs to be updated on the corresponding region $\mathbf{R}^\star$ determined at step 2 according to the tentative segmented image (20) obtained during step 1(b).
   - *combine best components*: here c_{n+1} is obtained by updating c_n on the 1 to 3 regions $\mathbf{R} \in \mathcal{P}_n$ which have been split to define $\mathcal{P}_{n+1}$

Remark 4 *It is of course possible to start the algorithm from any multiscalar partition $\mathcal{P}_1$ instead of three* 1*-partitions, with the restriction that $\mathcal{P}_1$ has to be a* vector partition *in case the* combine best components *multiscalar strategy is used.*

4 Numerical results

We illustrate in this section the numerical behavior of algorithms of sections 3.1.4 and 3.2.3 for the adaptive vector and multiscalar segmentations. The image d to be segmented is the picture of a colorful sculpture [24]. Its definition is 400 x 533 (213,200 pixels).

Note that the original image d is itself trivially a segmented image, with a number of regions equal, for a vector segmentation, to the number of distinct colors—here 55,505—, or equal, for each component for a multiscalar segmentation, to the number of distinct levels in the component—here the maximum of 256 levels for a 24-bit color image is reached for all components. Hence, if no constraint is set on the number of regions, all above adaptive segmentation algorithms will converge towards the original image d. But, though regions are introduced in our algorithms with parsimony at each iteration, there is for the time no mathematical proof that they will produce a segmented image c identical to d as soon as the number of regions in c becomes equal to that of the trivial segmentation of d. But c will for sure end up being identical to d at some point before the number of its regions equals the number of its pixels!

This convergence property is important as it ensures that by properly stopping the algorithm, one can control the compromise between the number of regions of the segmentation and the resemblance of the segmented image to the original one.

The software used to compute these numerical examples is still under development, but it will be available soon at `http://refinement.inria.fr/`.

4.1 Vector segmentation

We test first the vector segmentation algorithm 3.1.4 with two cutting strategies. Here the algorithm adds exactly one region at each iteration, so it is easy to specify a stopping criterion providing a segmentation into exactly n regions with uniform colors.

4.1.1 *Best in a family* cutting strategy

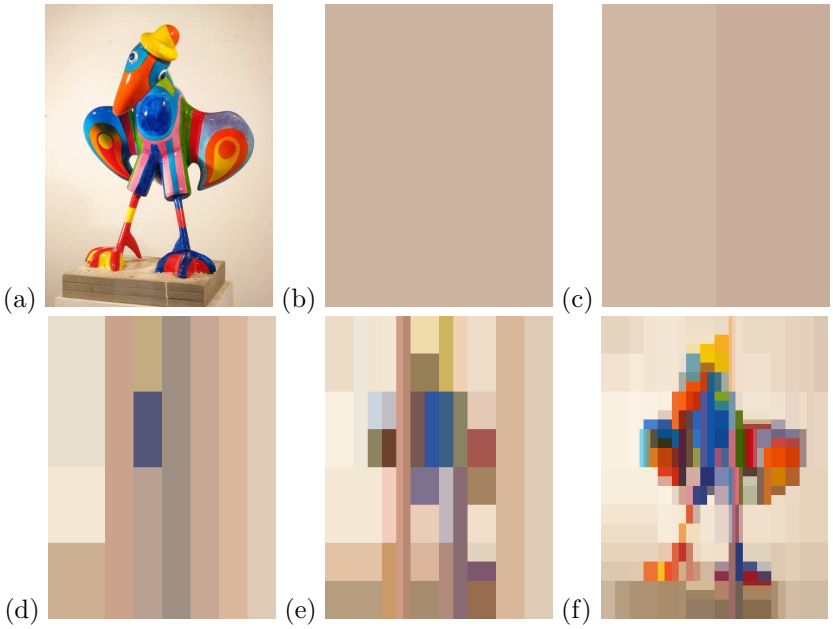

Figure 4: *Vector segmentation* with the *best in a family* cutting strategy. (a) is the image data to segment. Then, for each displayed image, we give the iteration number n, the number of regions n_{vr}, and the percentage of explained data τ.
(b) $n = 0$, $n_{vr} = 1$, $\tau = 65.0\%$;
(c) $n = 1$, $n_{vr} = 2$, $\tau = 65.1\%$;
(d) $n = 10$, $n_{vr} = 11$, $\tau = 68.7\%$;
(e) $n = 40$, $n_{vr} = 41$, $\tau = 73.4\%$;
(f) $n = 200$, $n_{vr} = 201$, $\tau = 82.2\%$.

The images shown in Figure 4 are obtained with a quite scarce predefined family of vertical and horizontal cuttings: regions are rectangles, and the only cuttings to be tested in each region are its horizontal and vertical divisions into two equal parts. For instance, the first cutting is vertical as shown in Figure 4(c), and all regions in Figures 4(d), 4(e) and 4(f) are rectangular. The exact indicators are used to rank all cuts.

The adaptive character of the algorithm is clear on images Figures 4(d), 4(e) and 4(f) where regions are extremely irregular. At each iteration, the cutting providing the largest decrease of the cost function is selected.

The convergence of the algorithm is very slow: the image Figure 4(f) obtained after 200 iterations is made of 200 monochrome rectangles, but it bears only a rough resemblance to the data. So it seems that the *best in a family* cutting strategy is not well suited for segmentation purpose—even though it remains a very good strategy for the general problem of vector parameter estimation.

4.1.2 *Overall best* cutting strategy

We present in Figure 5 the vector segmentation of the same image with the *overall best* cutting strategy. Here the shape of the regions are perfectly adapted to the image data: the image is already recognizable with only three regions, see Figure 5(d), and the algorithm is able to explain 86.8% of the data with as few as six color regions, see Figure 5(e). But it takes 41 color regions to explain up to 94.8% of the data. In any case, the intermediate images generated by the algorithm offer a large choice of segmented images to choose from.

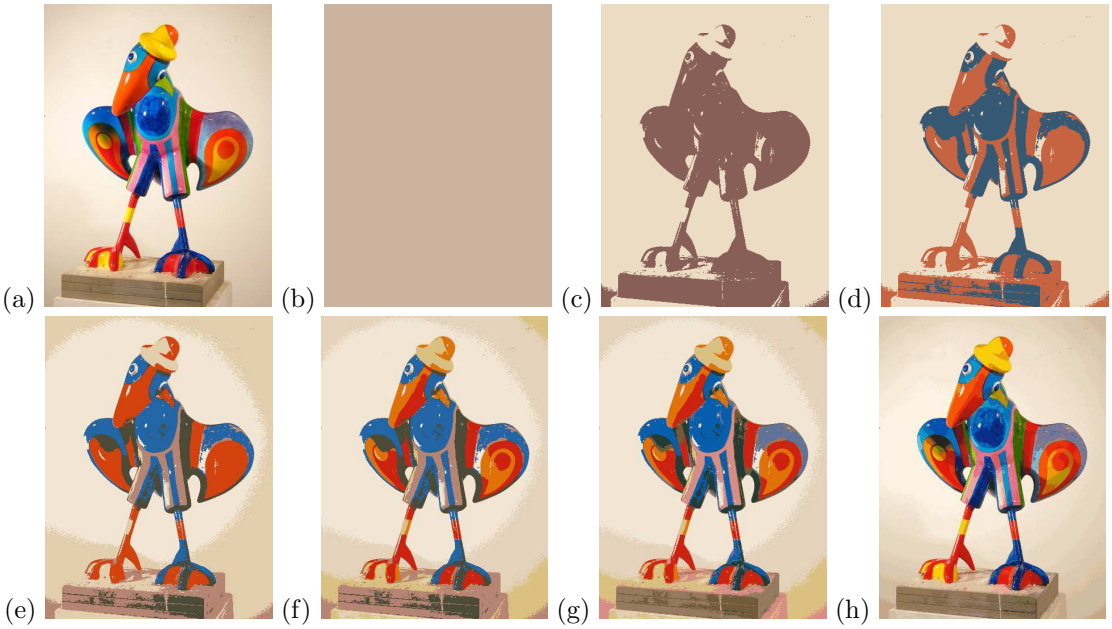

Figure 5: *Vector segmentation* with the *overall best* cutting strategy. (a) is the image data to segment. Then, for each displayed image, we give the iteration number n, the number of regions n_{vr}, and the percentage of explained data τ.
(b) $n = 0$, $n_{vr} = 1$, $\tau = 65.0\%$;
(c) $n = 1$, $n_{vr} = 2$, $\tau = 77.9\%$;
(d) $n = 2$, $n_{vr} = 3$, $\tau = 82.1\%$;
(e) $n = 5$, $n_{vr} = 6$, $\tau = 86.8\%$;
(f) $n = 7$, $n_{vr} = 8$, $\tau = 88.1\%$;
(g) $n = 10$, $n_{vr} = 11$, $\tau = 89.6\%$;
(h) $n = 40$, $n_{vr} = 41$, $\tau = 94.8\%$.

It is this configuration of the algorithm (vector segmentation with *overall best* cutting strategy) which was used to run the example of Figure 2. There also, as we have seen in section 2.4, the intermediate iterations are pertinent as they let us discover progressively the smaller perturbations.

In conclusion, the *overall best* cutting strategy seems to be well adapted to image segmentation purposes, so we have used it in all the numerical results below.

4.2 Multiscalar segmentation

For the general parameter estimation problem, multiscalar segmentation is a natural approach when each component of the vector parameter has a different physical interpretation. This is not the case for image segmentation, where the three color components have the same status. But one can expect that allowing a different partition for each component will allow to represent the image with a smaller number of regions for a same level of details. All tests in this section have been performed with the *overall best* cutting strategy.

4.2.1 The *best component only* multiscalar strategy

Here *one iteration* of the algorithm adds exactly *one scalar degree of freedom* to the component which produces the largest decrease of the objective function. Hence, one can expect to obtained a good representation of the original image with fewer component regions. This is confirmed by the comparison of Figures 6(f) and 5(g): the *best component only* multiscalar strategy explains

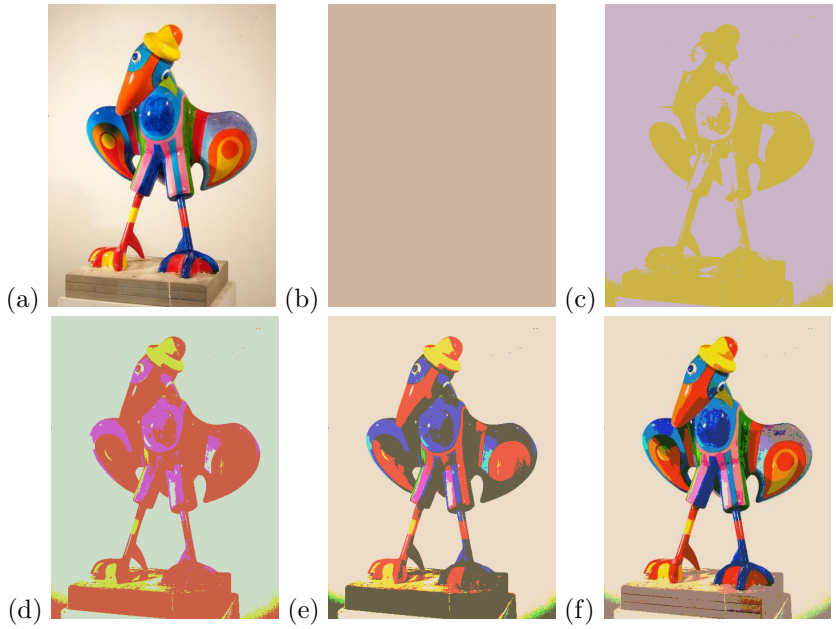

Figure 6: *Multiscalar segmentation* with the *overall best* cutting strategy and the *best component only* multiscalar strategy. (a) is the image data to segment. Then, for each displayed image, we give the iteration number n, the number of scalar regions n_{sr}, the resulting number of uniform color regions n_{vr}, and the percentage of explained data τ.
(b) $n = 0$, $n_{sr} = 3$, $n_{vr} = 1$, $\tau = 65.0\%$;
(c) $n = 1$, $n_{sr} = 4$, $n_{vr} = 2$, $\tau = 70.0\%$;
(d) $n = 2$, $n_{sr} = 5$, $n_{vr} = 4$, $\tau = 75.6\%$;
(e) $n = 3$, $n_{sr} = 6$, $n_{vr} = 8$, $\tau = 82.8\%$;
(f) $n = 6$, $n_{sr} = 9$, $n_{vr} = 26$, $\tau = 90.1\%$.

90% of the data with only 9 scalar regions (which generate 26 color regions), whereas the vector segmentation algorithm requires 11 color regions, which correspond to 33 component regions.

4.2.2 The *best component for each* multiscalar strategy

The control on the number of degrees of freedom is less precise with this algorithm, as *one iteration* adds exactly *three scalar degrees of freedom* instead of one for the previous algorithm.

Note that the images of Figures 7(d) and 6(f) coincide: this means simply that the three first iterations—after the three initialization ones—of the *best component only* algorithm have updated successively the R, G and B components. But when the number of iterations increases, it can be expected that the *best component only* algorithm will not cycle regularly through the R, G and B components as the *best component for each* does, and hence produce better images for a given number of component regions.

Because of the complete independence of the partitions for the R, G and B components, the number of color regions obtained by superimposition of the component regions blows up for these two algorithms, see Figure 7(f). This shows that the parameterization of the image by *component regions* instead of *color regions* allows a drastic reduction of the number of regions for a given image quality. In other words, it is possible with these multiscalar algorithms to obtain a segmented image with a large number of color regions, described by a much smaller number of component regions.

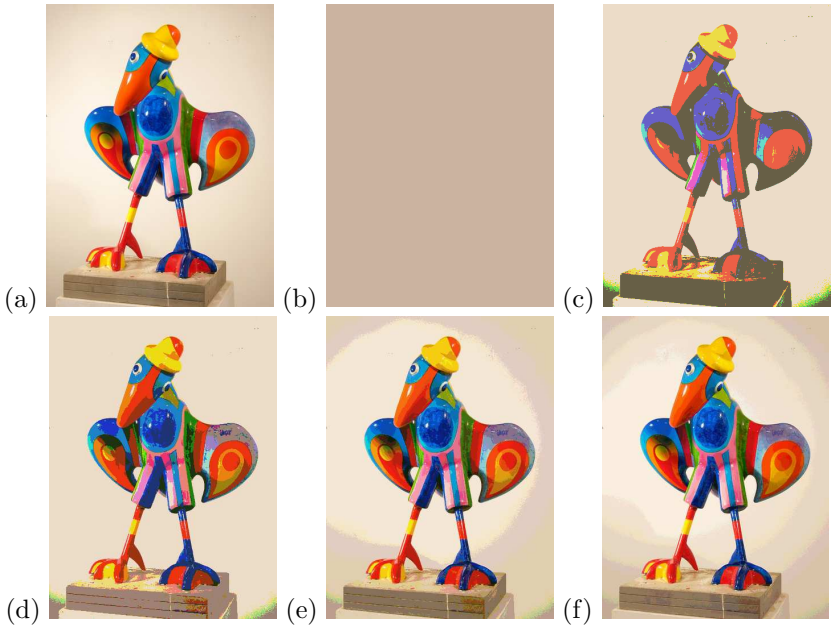

Figure 7: *Multiscalar segmentation* with the *overall best* cutting strategy and the *best component for each* multiscalar strategy. (a) is the image data to segment. Then, for each displayed image, we give the iteration number n, the number of scalar regions n_{sr}, the resulting number of vector regions n_{vr}, and the ratio of explained data τ.
(b) $n = 0$, $n_{sr} = 3$, $n_{vr} = 1$, $\tau = 65.0\%$;
(c) $n = 1$, $n_{sr} = 6$, $n_{vr} = 8$, $\tau = 82.8\%$;
(d) $n = 2$, $n_{sr} = 9$, $n_{vr} = 26$, $\tau = 90.1\%$;
(e) $n = 5$, $n_{sr} = 18$, $n_{vr} = 153$, $\tau = 94.7\%$;
(f) $n = 10$, $n_{sr} = 33$, $n_{vr} = 4581$, $\tau = 97.1\%$.

4.2.3 The *combine best components* multiscalar strategy

This multiscalar strategy produces in fact a vector segmentation, as we have seen in the last item of section 3.2.2. Hence, all current component images are segmented on the same partition, so we have omitted in Figure 8 the number n_{sr} of scalar regions, as it is given by $n_{sr} = 3n_{vr}$.

Comparison of Figures 8(d) and 8(e) with Figure 5(g) shows that, for a given level of accuracy, the *combine best components* multiscalar strategy tends to produce vector segmented images with a larger number of regions than the direct *vector segmentation algorithm*.

Also, the algorithm adds at each iteration a number of color regions between 3 and 7, i.e. between 9 and 21 scalar degrees of freedom, which makes it more difficult to control of the number of color regions.

So in a whole, the *combine best components* multiscalar strategy seems less performing for image *vector segmentation* than the direct vector segmentation algorithm 3.1.4.

Conclusion

Once recognized as a parameter estimation problem, for which the forward modeling map to invert is the identity, image segmentation may benefit from the theory of optimal control.

We have studied the specificities of this favorable case in the context of adaptive parameterization and we have proposed an *optimal adaptive (vector) segmentation algorithm*. This algorithm provides a finite sequence of segmented images with a number of *color regions* increasing by one at each iteration. The adaptive character shows itself in the fact that significant regions are extracted

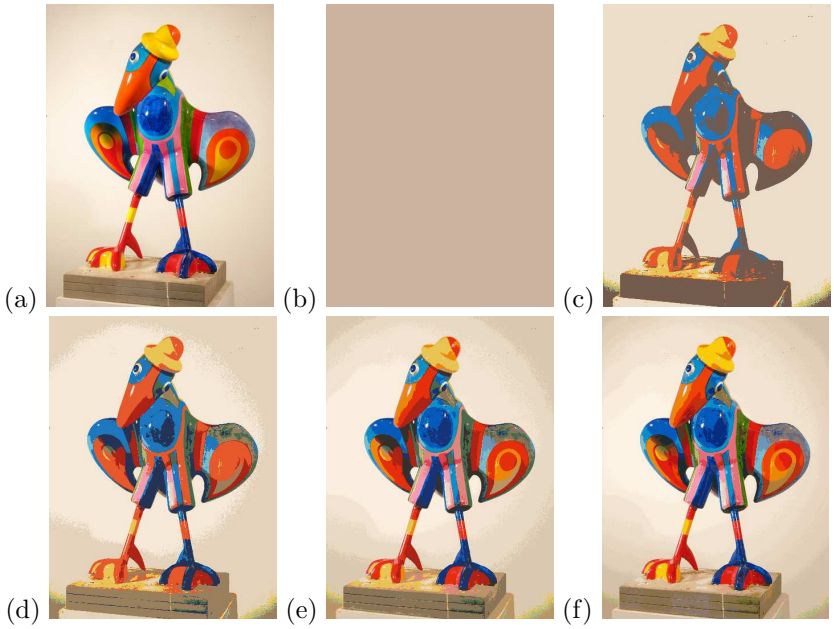

Figure 8: *Multiscalar segmentation* with the *overall best* cutting strategy and the *combine best components* multiscalar strategy. (a) is the image data to segment. Then, for each displayed image, we give the iteration number n, the number of uniform color regions n_{vr}, and the percentage of explained data τ.
(b) $n = 0$, $n_{vr} = 1$, $\tau = 65.0\%$;
(c) $n = 1$, $n_{vr} = 8$, $\tau = 84.1\%$;
(d) $n = 2$, $n_{vr} = 12$, $\tau = 88.3\%$;
(e) $n = 5$, $n_{vr} = 24$, $\tau = 91.4\%$;
(f) $n = 10$, $n_{vr} = 42$, $\tau = 94.1\%$.

from the very beginning of the process, and that all the intermediate steps of the sequence are meaningful. The optimal character indicates that the most significant regions are extracted first, and that the sequence minimizes the error with respect to the original image and terminates with a segmented image containing as many regions as distinct colors in the original image corresponding to the null error. This algorithm is best suited when a fine control of the number of color regions is desired.

The study has also exhibited a class of *multiscalar algorithms* that provide a distinct segmentation for each color component. Here the number of *component regions* increases by one or three at each iteration, but the number of resulting *color regions* blows up quickly, thus explaining a large percentage of the data. This algorithm is best suited when a desired percentage of the data has to be explained with a number of (component) regions as small as possible.

References

[1] C. Baillard, C. Barillot, and P. Bouthemy. Robust adaptive segmentation of 3D medical images with level sets. Technical Report 1369, Irisa, Rennes, France, November 2000.

[2] J.-C. Baillie. Segmentation. Technical report, ENSTA, 2003. Module D9, ES322 - Traitement d'Image et Vision Artificielle.

[3] H. Ben Ameur, G. Chavent, and J. Jaffré. Refinement and coarsening indicators for adaptive parameterization: application to the estimation of hydraulic transmissivities. *Inverse Problems*, 18:775–794, 2002.

[4] H. Ben Ameur, F. Clément, P. Weis, and G. Chavent. The multidimensional refinement indicators algorithm for optimal parameterization. *Journal of Inverse and Ill-Posed Problems*, 16:107–126, 2008.

[5] H. Ben Ameur and B. Kaltenbacher. Regularization of parameter estimation by adaptive discretization using refinement and coarsening indicators. *Inverse and ill posed problems*, 10(6):561–583, 2002.

[6] L. L. Bonilla, editor. *Inverse Problems and Imaging*, volume 1943 of *Lecture Notes in Mathematics (Fondazione C.I.M.E., Firenze)*. Springer, 2008.

[7] E. Borenstein and S. Ullman. Combined top-down/bottom-up segmentation. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 30:2109–2125, 2008.

[8] A. Chakraborty and J. S. Duncan. Integration of boundary finding and region-based segmentation using game theory. In *Information Processing in Medical Imaging*, pages 189–201. Kluwer, 1995.

[9] G. Chavent. *Nonlinear Least Squares for Inverse Problems*, volume XIV of *Scientific Computation*. Springer, 2010.

[10] J. Chen. Adaptive image segmentation based on color and texture. In *In Proceedings on International Conference on Image Processing (ICIP*, pages 789–792, 2002.

[11] H. D. Cheng, X. H. Jiang, Y. Sun, and J. Wang. Color image segmentation: advances and prospects. *Pattern Recognition*, 34(12):2259–2281, 2001.

[12] Jianping Fan, D.K.Y. Yau, A.K. Elmagarmid, and W.G. Aref. Automatic image segmentation by integrating color-edge extraction and seeded region growing. *Image Processing, IEEE Transactions on*, 10(10):1454–1466, Oct 2001.

[13] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions and the bayesian restoration of images. *IEEE PAMI*, 6, 1984.

[14] G. Koepfler, C. Lopez, and J. M. Morel. A multiscale algorithm for image segmentation by variational method. *SIAM Journal on Numerical Analysis*, 31(1):282–299, 1994.

[15] X. Muñoz, J. Freixenet, X. Cufí, and J. Martí. Strategies for image segmentation combining region and boundary information. *Pattern Recogn. Lett.*, 24(1-3):375–392, 2003.

[16] E. Nadernejad, S. Sharifzadeh, and H. Hassanpour. Edge detection techniques: Evaluations and comparisons. *Applied Mathematics Sciences*, 31:1507–1520, 2008.

[17] Y. Ohta, K. Kanade, and T. Sakai. Color information for region segmentation. *Comput. Graphics Image Process*, 13:222–241, 1980.

[18] C. Pachai, Y. M. Zhu, C. R. G. Guttmann, R. Kikinis, F. A. Jolesz, G. Gimenez, J. C. Froment, C. Confavreux, and S. K. Warfield. Unsupervised and adaptive segmentation of multispectral 3d magnetic resonance images of human brain: a generic approach. In *Proc. of the Fourth Internat. Conf. on Medical Image Computing and Computer-Assisted Intervention (MICCAI-2001)*, pages 1067–1074, 2001.

[19] N. R. Pal and S. K. Pal. A review on image segmentation techniques. *Pattern Recognition*, 26(9):1277–1294, 1993.

[20] T. N. Pappas. An adaptive clustering algorithm for image segmentation. *IEEE Transactions on Signal Processing*, 40(4):901–914, 1992.

[21] T. Pavlidis. *Structural Pattern Recognition.* Springer-Verlag, 1980.

[22] S. Philipp-Foliguet and L. Guigues. New criteria for evaluating image segmentation results. In *Proc. of IEEE Internat. Conf. on Acoustics, Speech and Signal Processing*, volume II, pages 109–112, 2006.

[23] M. Pietikäinen, A. Rosenfeld, and I. Walter. Split-and-link algorithms for image segmentation. *Pattern Recognition*, 15(4):287–298, 1982.

[24] Les Popille. Gros bec à chapeau jaune en short. `http://www.popille.fr/archives/-nondispo/18-GrosBecAChapeauJauneEnShort.htm`, 2004.

[25] C. Rosenberg, K. Chehdi, and C. Kermad. Adaptive segmentation system. In *Proc. of the 5th Internat. Conf. on Signal Processing*, volume 2, pages 918–921, 2000.

[26] W. Skarbek and A. Koschan. Colour image segmentation - a survey. Technischer Bericht 94-32, Technical University of Berlin, 1994.

[27] L. Spirkovska. A summary of image segmentation techniques. Technical Memorandum 104022, NASA, Ames Research Center, Moffett Field, California, 1993.

[28] S. Tominaga. Color image segmentation using three perceptual attributes. *IEEE Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 626–630, 1986.

[29] A. Tremeau and N. Borel. A region growing and merging algorithm to color segmentation. *Pattern Recognition Lett*, 13:561–568, 1997.

[30] T. Uchiyama and M.A. Arbib. Color image segmentation using competitive learning. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 16(12):1197–1206, 1994.

[31] X. Wu. Adaptive split-and-merge segmentation based on piecewise least-square approximation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 15(8):808–815, 1993.

Centre de recherche INRIA Paris – Rocquencourt
Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Centre de recherche INRIA Bordeaux – Sud Ouest : Domaine Universitaire - 351, cours de la Libération - 33405 Talence Cedex
Centre de recherche INRIA Grenoble – Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier
Centre de recherche INRIA Lille – Nord Europe : Parc Scientifique de la Haute Borne - 40, avenue Halley - 59650 Villeneuve d'Ascq
Centre de recherche INRIA Nancy – Grand Est : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex
Centre de recherche INRIA Rennes – Bretagne Atlantique : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex
Centre de recherche INRIA Saclay – Île-de-France : Parc Orsay Université - ZAC des Vignes : 4, rue Jacques Monod - 91893 Orsay Cedex
Centre de recherche INRIA Sophia Antipolis – Méditerranée : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399