Image segmentation using a generic, fast and non-parametric approach

Christophe Fiorio, Richard Nock

Image Segmentation Using a Generic, Fast and Non-Parametric Approach

Christophe Fiorio, Richard Nock
LIRMM, 161 rue Ada, F-34790 Montpellier Cedex 5

Abstract:

In this paper, we investigate image segmentation by region merging. Given any similarity measure between regions, satisfying some weak constraints, we give a general predicate for answering if two regions are to be merged or not during the segmentation process. Our predicate is generic and has six properties. The first one is its independance with respect to the similarity measure, that leads to a user-independant and adaptative predicate. Second, it is non-parametric, and does not rely on any assumption concerning the image. Third, due to its weak constraints, knowledge may be included in the predicate to fit better to the user's behaviour. Fourth, provided the similarity is well-chosen by the user, we are able to upperbound one type of error made during the image segmentation. Fifth, it does not rely on a particular segmentation algorithm and can be used with almost all region-merging algorithms in various application domains. Sixth, it is calculated quickly, and can lead with appropriated algorithms to very efficient segmentation.

Keywords: Vision and Image Processing, AI algorithms, Machine Learning.

1 Introduction

In the early stage of a vision process it is necessary to isolate "objects" in the scene before recognizing them. This step in the analyzing process is called image segmentation. There are several approaches to this problem [1, 2], namely threshold techniques ([3] for example), boundary (such as Canny-Deriche edge detector [4, 5]) and regions based methods ([6] and [7] for more recent results), and hybrid techniques (as for example [8, 9]) combining the two last methods. In this paper we are concerned by region segmentation of intensity (*i.e.* grey levels) images and more precisely region merging. The goal of region segmentation is to divide the image I in regions $R_1, \ldots, R_n$ such that $I = \bigcup_{i=1}^{n} R_i$, $R_i \bigcap R_j = \emptyset$ if $i \neq j$, each region being connected and satisfying an homogeneity criterion. Region merging builds up regions by combining smaller regions, pixels being taken as primary regions, using a predicate $\mathcal{P}$ as a similarity test.

The quality of a segmentation is strongly dependent on the way one answer to a simple question: "should (or should not) be merged two regions ?" Albeit a user can answer to this question, it is hard to give further steps in the definition of a segmentation algorithm. The first reason comes from the difficulty to quantify a mathematical notion of similarity between regions, which is a part of the user's judgment, but rather implicit and semantic dependent in his case. Second, the use of this similarity for answering to the latter question is not straightforward, since it implies defining a mapping from the similarity values to the Boolean values $\{yes, no\}$. Third, the main bottleneck is obviously the design of the whole segmentation algorithm which uses such predicate.

Many segmentation algorithms use statistical or mathematical techniques relying on hypotheses made *a priori* on the image [10, 11, 12]. In that case, the time spent for the calculations can be very important. Furthermore, many of these hypotheses are parametric, introducing distributional assumptions on the image. *A contrario*, Other approaches focuse on algorithmic properties of the image [13, 14], leading to efficient segmentation algorithms. Our approach differs from all these ones in that it is based on the combination of statistical and algorithmic properties of a criterion used for the segmentation. Statistical properties are based on the absence of any parametric hypotheses on the image. Algorithmic properties are based on a quick calculation (constant-time) of the criterion, that can lead to optimal image segmentation (both linear-time and linear-space).

In this paper we give a generic non parametric predicate for image segmentation based on "distance" between regions. It is generic in the sense that it is independent from the chosen distance and independent of a particular region segmentation algorithm. Furthermore, provided the user can give a calculable "distance" relating his distance, a part of the error made during the segmentation process can be controlled. Such errors can arise *e.g.* when merging two regions that represent in fact distinct objects. Moreover, it satisfies the desirable property of being calculable in constant time for a broad class of "distances". So used with an adequate algorithm we can achieve a linear-time and linear-space segmentation process. At last as shown at the end of Section 3.2 our predicate is auto-adaptative since, except for a risk parameter, the user doesn't tune others parameters and the limit value of the similarity is set automatically

0-7803-5214-9/98/$10.00

according to the risk and the image.

In the next section we provide basic definitions, as well as the algorithm `mergeSquare` we use in our application. Then, we present in the following section our generic predicate. In the same section, we discuss about the fast calculation of the predicate criteria. In the last section, we present some similarity measures we used, and some experimental results we obtained using them. We also present an application where some knowledge is put in our criterion, and the quality of the segmentation obtained. We leave in the appendix some theoretical remarks on the error reduction process that occurs during segmentation, using our criterion.

2 Preliminaries

2.1 Definitions

An image I with r rows and c columns is an $r \times c$ matrix taking its entries in $\{1, ..., g\}$ where g is an integer representing the number of grey levels of I. A *region* of I is a non-empty, 4-connected[1] subset of 2^I. Two distinct regions of I are *adjacent* if and only if (iff for short) their union is still a region. A *segmentation predicate* is a predicate $\mathcal{P}(R, R')$ taking as parameters two adjacent regions R and R' and answering "Y" or "N" whether or not R and R' should be merged during the segmentation process. The segmentation predicate uses a *criterion* $\mathtt{C}(R, R')$ measuring the "proximity" of regions R and R'. Formally, $\mathtt{C}$ is only required to satisfy the following conditions:

- $\mathtt{C}$ is an application from $2^I \times 2^I$ to $\mathbb{R}$, having finite values,
- $\mathtt{C}$ is symmetric : $\forall R, R' \in 2^I, \mathtt{C}(R, R') = \mathtt{C}(R', R)$.

Intuitively, the smaller $\mathtt{C}(R, R')$, the more likely R and R' are to be merged according to $\mathcal{P}$. Since it depends on $\mathtt{C}$, we can rewrite $\mathcal{P}$ as $\mathcal{P}_{\mathtt{C}}$.

A basic example of criterion $\mathtt{C}(R, R')$ is the mean difference $\mathtt{MD}(R, R')$:

$$\left| \frac{1}{|R|} \sum_{(x,y)\in R} I(x,y) - \frac{1}{|R'|} \sum_{(x,y)\in R'} I(x,y) \right|$$

and a segmentation predicate using MD would be

$$\mathcal{P}_{\mathtt{MD}}(R, R') = \text{“}\mathtt{MD}(R, R') < 20\text{”}$$

[1]Two pixels are 4-adjacent iff they share an edge. A region is 4-connected iff given any two pixels p and q of it, there exists a path of pixels in the region, $p_1 = p, p_2, ..., p_k = q$, such that $\forall i \in \{1, ..., k-1\}$, p_i is 4-adjacent to p_{i+1}.

$\mathcal{P}_{\mathtt{MD}}(R, R')$ is read as "if $\mathtt{MD}(R, R') < 20$, then R and R' can be merged into one region". The design of $\mathcal{P}_{\mathtt{C}}$ is not straightforward. In particular, it depends on the choice of $\mathcal{C}$. In our example, the constant "20" we give for $\mathcal{P}_{\mathtt{MD}}$ is chosen arbitrarily ; it is *a priori* difficult to know if this constant is better than another one. Note also that the design of $\mathcal{P}$ is not straightforward if we are allowed to choose arbitrary $\mathtt{C}$.

2.2 The algorithm `mergeSquare`

In order to achieve linear-time and linear-space region segmentation algorithm we will use for illustration purpose the `mergeSquare` algorithm (see next page) presented in [14]. This algorithm is based on a union-find structure which represents regions by trees of pixels belonging to the same region (see [15] for more information on union-find algorithms and [16, 14] for application to image processing). It proceeds by a quad-tree-like scan of the pixels of the image, and for each pixel on the boundary of a square the algorithm checks if the region it belongs to can be merge with the region to which belongs the pixel on the other side of the boundary. The merge (the *union*) of two regions is done by linking the root of the tree of one region to the other one. Note that to identify the region a pixel belongs to, it is sufficient to check the root of its tree, the *find* operation allowing us to find quickly the roots of trees. In Algorithm 1, the image is assumed to be a square matrix, and the four sub-squares of a given matrix are called $S_{NW}, S_{NE}, S_{SW}, S_{SE}$.

To be linear this algorithm requires to use *constant-time*

Algorithm 1: `mergeSquare(`S,k`)`

```
Input: an integer k and a square S
if k = 0 then return ;
h⁻ = 2^(k-1) ; h⁺ = 2^(k-1);
for D=NW to SE do mergeSquare(S_D, k − 1) ;
for i = 0 to 2^k do
    left = Find(S[i, h⁻]) ;
    right = Find(S[i, h⁺]) ;
    if P(left,right) then Union(left,right) ;
for i = 0 to 2^k do
    up = Find(S[h⁻, i]) ;
    down = Find(S[h⁺, i]) ;
    if P(up,down) then Union(up,down) ;
```

predicates, i.e. predicates that are calculated in constant-time at each stage of the process. Note that this implies that parameters required for the calculus (such as sums of grey levels and cardinal of a region for the mean criterion for example) must be updated in constant time after a union is done.

3 The generic predicate

3.1 Segmentation criterion and associated predicate

The idea underlying our predicate criterion is the following : two adjacent regions R and R' can be merged during the segmentation process iff

- $\mathrm{C}(R, R')$ is sufficiently small, and
- $\mathrm{C}(R, R')$ is not greater than many $\mathrm{C}(R'', R''')$, where $R'' \neq R'''$ and R'', R''' cover the regions of I.

We create a "meta-criterion" $F(\mathrm{C}(R, R'))$, using a function $F(x)$ whose domain is $\mathbb{R}$, taking its values in $[0, 1]$. It takes place between the classic criterion C and the predicate $\mathcal{P}_{\mathrm{C}}$. Given any user-defined criterion C, F is defined as

$$F(x) = \frac{\left|\left\{ R'', R''' \in I : \left[\begin{array}{l} R'' \neq R''' \\ \mathrm{C}(R'', R''') \leq x \end{array} \right. \right\}\right|}{|\{R'', R''' \in I : R'' \neq R'''\}|} \quad (1)$$

$F(x)$ is the ratio of couple of regions for which the value $\mathrm{C}(R'', R''')$ is smaller than that of $\mathrm{C}(R, R')$. Since we do not make assumptions on the image, we do not suppose that regions R'' and R''' are linked. The predicate criterion $\mathcal{P}_{\mathrm{C}}$ is now the following:

$$\mathcal{P}_{\mathrm{C}}(R, R') \equiv \text{“}F(\mathrm{C}(R, R')) < \alpha_1\text{”} \quad (2)$$

where α_1 is a risk associated to $\mathcal{P}_{\mathrm{C}}$ (generally, $3\% \leq \alpha_1 \leq 10\%$). α_1 corresponds to the (small) percent of regions that can be merged.

3.2 Representation of $F(x)$

We suppose that the image I is segmented, and contains a set S of regions (at the beginning of the algorithm, each pixel defines one element of S). It is convenient for our purpose to present $F(x)$ as the integral of an estimated density $f(x)$. We build the function $f(x)$ in the following way. Let $\{v_0, v_1, ..., v_k\}$ stands for the set of possible values of $\mathrm{C}(R'', R''')$, $R'' \neq R'''$ and $R'', R''' \in S$. Then $\forall i \in \{1, 2, ..., k\}, \forall x \in]v_{i-1}, v_i]$,

$$f(x) = \frac{\left|\left\{ R'', R''' \in I : \left[\begin{array}{l} R'' \neq R''' \\ \mathrm{C}(R'', R''') = v_i \end{array} \right. \right\}\right|}{(v_i - v_{i-1}) \times |\{R'', R''' \in I : R'' \neq R'''\}|}$$

and $\forall x \in [v_0 - 1, v_0]$,

$$f(x) = \frac{\left|\left\{ R'', R''' \in I : \left[\begin{array}{l} R'' \neq R''' \\ \mathrm{C}(R'', R''') = v_0 \end{array} \right. \right\}\right|}{|\{R'', R''' \in I : R'' \neq R'''\}|}$$

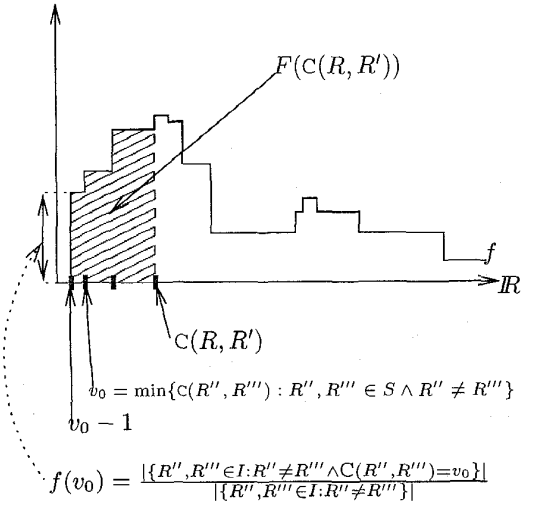

Figure 1: The construction of f and a representation of $F(\mathrm{C}(R, R'))$.

We also set

$$\forall x \in]-\infty, v_0 - 1[\cup]v_k, +\infty[, f(x) = 0$$

Figure 1 shows the construction of f as well as a representation of $F(\mathrm{C}(R, R'))$ that we justify now. $\forall i \in \{0, 1, ..., k\}$, $F(v_i)$ can be calculated using the following formula: $F(v_i) = \int_{-\infty}^{v_i} f(z)dz$. Indeed, by setting $v_{-1} = v_0 - 1$, we obtain

$$\begin{aligned} &\forall 0 \leq i \leq k, \int_{-\infty}^{v_i} f(z)dz = \sum_{j=0}^{i} (v_j - v_{j-1}) f(v_j) \\ &= \sum_{j=0}^{i} \frac{\left|\left\{ R'', R''' \in I : \left[\begin{array}{l} R'' \neq R''' \\ \mathrm{C}(R'', R''') = v_j \end{array} \right. \right\}\right|}{|\{R'', R''' \in I : R'' \neq R'''\}|} \\ &= \frac{\left|\left\{ R'', R''' \in I : \left[\begin{array}{l} R'' \neq R''' \\ \mathrm{C}(R'', R''') \leq v_i \end{array} \right. \right\}\right|}{|\{R'', R''' \in I : R'' \neq R'''\}|} = F(v_i) \end{aligned}$$

The integral formula for the calculation $F(x)$ shows the monotonocity of F, because f takes no negative value. Given two adjacent regions R, R' of S, note that $F(\mathrm{C}(R, R'))$ is just the area delimited between f and three lines whose equations are $y = 0$, $x = v_0 - 1$ and $x = \mathrm{C}(R, R')$.

The signification of $\mathcal{P}_{\mathrm{C}}$ can be illustrated by Figure 2 (it also shows a function f^* whose role is explained in in the appendix. It it represented here for convenience). The decision to merge (or not to merge) two regions is taken by comparing $F(\mathrm{C}(R, R'))$ and α_1, the two hatched areas in Figure 2. Equivalently, since $F(x)$ is monotonous, the decision consists in comparing $\mathrm{C}(R, R')$ and x_{α_1}, where x_{α_1} satisfies $F(x_{\alpha_1}) = \alpha_1$. In the example of Figure 2, since $F(\mathrm{C}(R, R')) < \alpha_1$ (equivalently, $\mathrm{C}(R, R') < x_{\alpha_1}$), $\mathcal{P}_{\mathrm{C}}(R, R')$ returns "Y" and regions R and R' are merged.

Note that the limit value x_{α_1} is not set by the user. Its calculation depends on the risk α_1, as well as on the image.

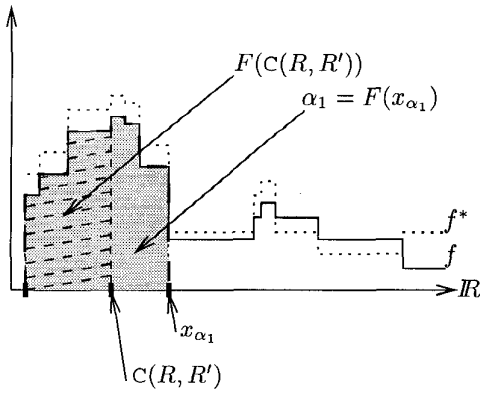

Figure 2: According to $\mathcal{P}_{\mathtt{C}}(R,R')$, regions R and R' can be merged.

Even when setting α_1 to a constant value, different images shall lead to different value of x_{α_1}. In this sense we can say that our predicate is *auto-adaptative*. This last point is the most important. We put in appendix some remarks concerning the limitation of the error during the segmentation process, that are not needed for our application purposes. Of greater importance is the way to calculate rapidly $F(x)$.

3.3 Rapid calculation of F

We define a predicate $\mathcal{P}_{\mathtt{C}}$ (resp. a criterion $\mathtt{C}$) to be a *constant-time predicate* (resp. *constant-time criterion*) iff its calculation from two regions R and R' can be done in constant time. This constant-time property is very important, as $\mathtt{C}$ and $\mathcal{P}_{\mathtt{C}}$ are accessed each time the merging of two regions is tested. Reducing as small as possible the complexity of their calculation is a proof of rapidity for the segmentation algorithm.

Obtaining a constant-time predicate $\mathcal{P}_{\mathtt{C}}$ implies necessarily to have

1. a constant-time criterion $\mathtt{C}$, and
2. a constant-time calculation of $F(\mathtt{C}(R,R'))$.

We propose in the following section criteria $\mathtt{C}$ that are constant-time, satisfying the first condition. Note however that the calculation of $F(\mathtt{C}(R,R'))$ requires more than constant time : in the worst-case, it is linear in the size of the image. In our calculations, we replace $F(\mathtt{C}(R,R'))$ by an estimator $\hat{F}(\mathtt{C}(R,R'))$ having the constant-time property we look for. It is calculated on the basis of a set P_t of t (constant) random observations of pairs of regions, over the regions of I. $\hat{F}(\mathtt{C}(R,R'))$ equals

$$\frac{\left|\left\{\{R'',R'''\} \in P_t : \left[\begin{array}{l} R'' \neq R''' \\ \mathtt{C}(R'',R''') \leq \mathtt{C}(R,R') \end{array}\right\}\right|}{t}$$

We have [17]:

$$P_r\left(\left|\hat{F}(\mathtt{C}(R,R')) - F(\mathtt{C}(R,R'))\right| < \sqrt{\frac{1}{2t}\log\frac{2}{\alpha}}\right) > 1-\alpha$$

Let $\alpha = \alpha_1$. Let $\alpha_2 = \sqrt{\frac{1}{2t}\log\frac{2}{\alpha_1}}$, where α_2 is a fixed risk parameter. Solving for t leads to

$$t = \frac{1}{2(\alpha_2)^2}\log\frac{2}{\alpha_1} \tag{3}$$

which is a constant for any fixed values of α_1 and α_2.

An illustration of the consequence of constant-time calculation of $\mathcal{P}_{\mathtt{C}}$ is given by our choice of the algorithm `mergeSquare` to implement our predicate. Indeed, we have [14]:

Theorem 1 *If $\mathcal{P}_C$ is a constant-time predicate, `mergeSquare` is linear-time and linear-space with respect to the size of I.*

4 Application

4.1 Specialization and quality improvement of the predicate criterion

The basis of the choice to merge two regions relies on the construction of a distribution f. This distribution is not necessarily built on the image that is currently being segmented:

- it can be another image, in the same domain, chosen because it has better quality (contrast, luminosity, etc...),
- it can be a little part of an image, chosen because it presents important details that should be treated carefully when seen in the segmented image,
- it can be an image taken from a collection of presegmented images, in relevant domains. For example, the segmentation of a boat can be done by constructing f from a presegmented boat image. Similarly, the medical segmentation of some organ can be done using a presegmented image of this organ, or similar images,
- f can be constructed using $i > 1$ images, putting $f = \frac{1}{i}\sum_{j=1}^{i} f_j$ where f_j is the distribution constructed from the j^{th} image.

These three points point out the inner possibility of putting knowledge, and machine learning in our predicate.

Given one image (or more) to build f, the predicate $\mathcal{P}_C$ can also be specialized in many ways to integrate knowledge on the image being segmented, in order to fit as precisely as possible to the users behaviour. This is equivalent to constructing conditional densities for f. We can define three types of specializations. The first ones are syntaxic specializations. Among these are all comparisons involving regions characteristics, such as

- using only regions whose grey-levels are limited, or with more general functions involving grey-levels of images,
- considering regions whose shape is constrained, or belongs to some class,
- considering regions not violating some constraint, such as the absence of edges between them if the image has been preprocessed using an edge-detection filter.

More generally, syntaxic specializations can be of any type, provided we can give some precise predicate for their calculation. Figure 3 gives an example of such predicate that can be added to the calculation of $F(x)$ in equation 1.
The second type of specialization are predicates involving a semantic knowledge of the image segmented, or (more likely) of benchmark images used to build f, as it is precised in the beginning of this section. For example, We may want to merge only regions that can be part of an organ, that would be described on a short benchmark of presegmented images. This makes the segmentation "object-oriented" since it tends to isolate large regions that are likely to belong to the searched object (in our example, an organ). Figure 3 gives another examples of semantic specialization for our predicate criterion.
Finally, the third type of possible specializations of our predicate, as shown in figure 3, relies on both syntaxic and semantic specializations.

Specializations of our predicate criterion drops downs to defining an *homogenity* predicate H, used to constrain the distribution f to more interesting distributions. The calculation of $F(x)$ can therefore be modified, and in the general case, we can rewrite equation 1 by:

$$F(x) = \frac{\left| \left\{ R'', R''' \in I : \begin{bmatrix} R'' \neq R''' \\ \mathtt{H}(R'', R''') = \mathtt{True} \\ \mathtt{C}(R'', R''') \leq x \end{bmatrix} \right\} \right|}{\left| \begin{array}{l} \{R'', R''' \in I : \\ R'' \neq R''' \wedge \mathtt{H}(R'', R''') = \mathtt{True}\} \end{array} \right|} \quad (4)$$

For example, in the case where we use an edge-detector to prevent merging regions separated by an edge, $\mathtt{H}(R'', R''')$ would be "there is no edge between R'' and R'''". In the case where we want to limit the grey-levels of merged regions, $\mathtt{H}(R'', R''')$ could be "the average grey-levels of R'' and R''' are not greater than x".

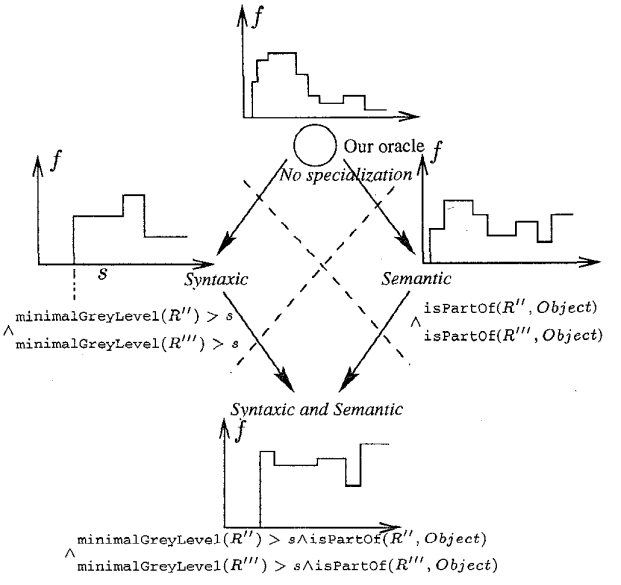

Figure 3: Possible specializations of our criterion.

4.2 Examples of constant-time criteria C

We now give some basic criteria C that were used to perform our experiments. They are all constant-time criteria. Their calculation is based on the following quantities defined for any region R, whose update is constant-time when merging two regions: the number of pixels of R ($|R|$), the grey level average of R ($\overline{R}$), the maximal grey level of R ($\max_R$) and the minimal grey level of R ($\min_R$). Their update when merging two regions R and R' in a single region $R \cup R'$ is performed in constant-time, since we have $|R \cup R'| = |R| + |R'|$ and $\overline{R \cup R'} = (|R|\overline{R} + |R'|\overline{R'})/(|R| + |R'|)$ and $\max_{R \cup R'} = \max\{\max_R, \max_{R'}\}$ and $\min_{R \cup R'} = \min\{\min_R, \min_{R'}\}$.

The first criterion is the mean difference MD that we can define as $\mathtt{MD}(R, R') = |\overline{R} - \overline{R'}|$. The second one is the range RA which traduces how much the histogram of grey levels in the union $R \cup R'$ flattens:

$$\begin{aligned} \mathtt{RA}(R, R') &= \max_{(x,y) \in R \cup R'} I(x, y) - \min_{(x,y) \in R \cup R'} I(x, y) \\ &= \max\{\max_R, \max_{R'}\} - \min\{\min_R, \min_{R'}\} \end{aligned}$$

The third one is the between-regions error BR, which is used in discriminant analyses:

$$\mathtt{BR}(R, R') = |R|(\overline{R} - \overline{R \cup R'})^2 + |R'|(\overline{R'} - \overline{R \cup R'})^2$$

We have tested other parameters, including variations in standard error of means, variations in the sums of grey levels, but the results were never better than those provided by these three criteria.

4.3 Basic experiments

Due to the lack of space, Figures 4, 6 present only a few experiments that were carried out, on three original benchmark images presented in figure 5. Our whole segmentation algorithm includes repeating N times the algorithm `mergeSquare` on each image, in order to decrease the region number. Each time, N was chosen small, never exceeding 15. Parameters α_1 and α_2 were tested with different values, using each of the criteria `MD`, `RA` and `BR`. Each result is presented by a four-tuple $(N, \alpha_1, \alpha_2, C)$. Our experiments show the interest of considering different criteria, leading to segmentations that behave quite differently. This difference is to be seen not on different qualities, but on different varieties. Indeed, unless we try bad segmentation criteria, the segmentations had all similar quality, but highlighted different part of the images, this being important since various domain studies shall focuse on various aspects of images and objects. Furthermore, we carried experiments setting high values α_1 and α_2 (one is showed with both values set to .20), and the quality of the segmentation obtained for these high values accounts for the robustness of our criterion. Real-time calculation never exceeded five minutes for the biggest images, for a non-optimized C program ran on a Pentium Pro 180. Processing time for 256×256 images never exceeded 1 minute.

4.4 An example of specialization of our criterion

We present an application of a syntaxic specialization of our criterion on the image `lena`.

Figure 5 shows the original image. We have first isolated a small part of the image, known as being very difficult to segment : the top of the hat. Figure 6 shows this part. We then have produced a first segmentation using an error criterion, which we denote `ER`:

$$\mathrm{ER}(R, R') = \sum_{(x,y)\in R\cup R'} \left(I(x,y) - \overline{R\cup R'}\right)^2$$

It is straightforward to show that this criterion is also constant-time, provided we keep for each region the sum of squares of grey levels (which also updates in constant time). We did not use separately this criterion like the three others, since it gave poor results for segmentations like in the preceeding section (the number of regions is generally greater than for the three other parameters). We have executed our criterion building f not on `lena`, but on the small part of the hat. This produces an histogram which is highly concentrated around the difficult values, and therefore leads to keeping almost perfectly the hat in the final image. The errors due to lena's hair and her hat's feathers are greatly reduced (a rectangle shows the area of interest on figure 6).

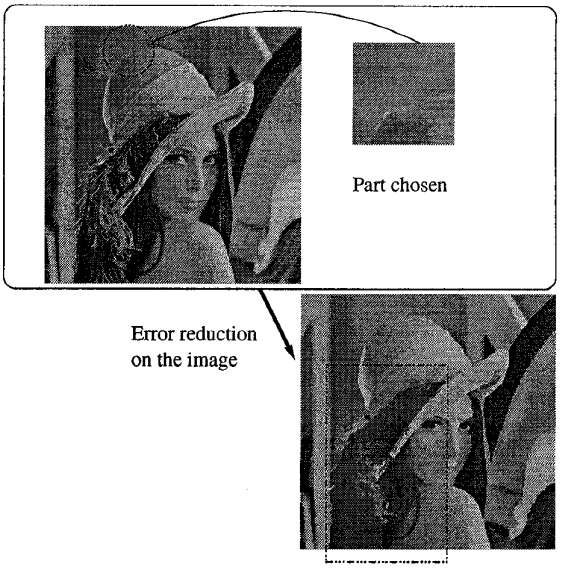

Figure 6: The error reduction in `lena`, with parameters $(20, .80, .10, \mathrm{ER})$.

This latter image obtained is then segmentated, but using to build f a very noisy image where greylevels are similar to `lena`. The final result is presented in figure 7 (left image). It contains very few regions (less than $0,06\%$ of the original number of regions), and many intuitive regions of the original image are kept almost intact. For comparison, the right image in figure 7 gives the result of a segmentation using the same criterion `MD`, chosen because it led to good decreasing of the regions number in lena's hair and hat's feathers. It shows that the segmentation using the preprocessed image of figure 6 leads effectively to reducing the error on regions such as lena's hair and hat's feathers.

5 Conclusion

We have presented in this paper a generic non parametric predicate to be used in region segmentation algorithms. It is generic since it for each distance criterion chosen we get a new predicate. This choice of a distance is critical since it determines the final result. The important thing to note is that, provided the distance criterion chosen fit exactly a user criterion segmentation, part of the predicate error with respect to the user is controlled. The stating that the similarity measure is well chosen by the user is strong, but due to the weak constraints on C, many knowledge can be

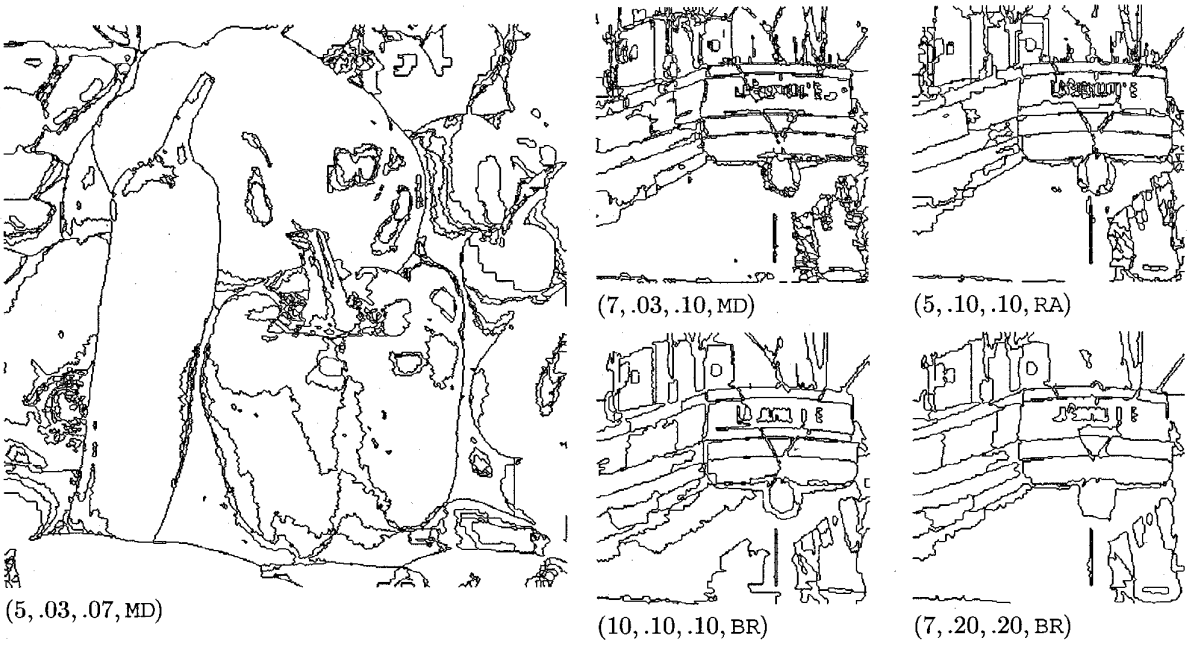

Figure 4: Segmentation produced with various parameters setting, see text.

Figure 5: Original images (resp. peppers, cornouaille, lena).

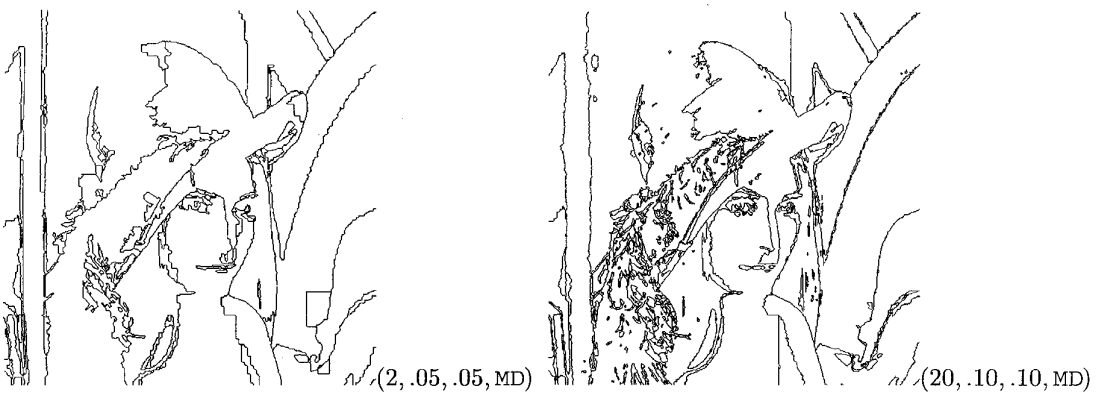

Figure 7: Final segmentations with `lena`, with error reduction (left), or without (right).

included to improve its behaviour.

Moreover our predicate doesn't need to tune parameters in order to get good result. It is only necessary to set a risk parameter, this risk being generally fixed within 3% to 10%. We claim that it is auto-adaptative since, providing the risk paramater is fixed, a limit value on the distance function will be set automatically according to the distribution of the image. This limit value will determine the similarity notion between regions in the image. It will be different in another image.

At last, if the chosen distance function is a constant-time predicate, we ensure that our predicate will be a constant-time predicate. This allows to design efficient segmentation algorithms. In Section 4 we have given some results showing good segmentations with simple distance criteria. This allow us to expect very good results in particular applications where knowledge can be included in the distance criterion.

References

[1] R. M. Haralick and L. G. Shapiro, "Survey: Image segmentation techniques," *Computer Vision, Graphics, and Image Processing.*, vol. 29, pp. 100–132, 1985.

[2] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*, vol. 1 and 2. Addison-Wesley Publishing Company, Inc., June 1992.

[3] T. Quiger, P. Miché, and R. Debrie, "Segmentation by auto-adaptative thresholding," in *Proceedings of the 6th Int. Conf. on Image Analysis and Processing*, (Villa Olmo, Como, Italy), pp. 34–42, 4–6 September 1991. Progress in Image Analysis and Processing II.

[4] J. Canny, "A computational approach to edge detection," *IEEE trans. Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698, 1986.

[5] R. Deriche, "Separable recursive filtering for efficient multi-scale edge detection," in *Int. Workshop on Machine Vision & Machine Intelligence*, (Tokyo), December 1987.

[6] S. W. Zucker, "Survey: Region growing: Childhood and adolescence," *Computer Vision, Graphics, and Image Processing.*, vol. 5, pp. 382–399, 1976.

[7] Y.-L. Chang and X. Li, "Fast image region growing," *Image and Vision Computing*, vol. 13, pp. 559–571, September 1995.

[8] J. Benois and D. Barba, "Image segmentation by region-contour cooperation for image coding," in *Proc. 11^th Int. Conf. on Pattern recognition (ICPR92)*, pp. 331–334, 1992.

[9] J.-P. Gambotto, "A new approach to combining region growing and edge detection," *Pattern Recognition Letters*, 1993.

[10] S. C. Zhu and A. Yuille, "Region competition : Unifying snakes, region growing, and bayes/MDL for multiband image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 884–900, 1996.

[11] S. Dugelay, C. Graffigne, and J.-M. Augustin, "Segmentation of multibeam acoustic imagery in the exploration of the deep-sea bottom," in *Int. Conf. on Pattern Recognition*, pp. 437–446, 1996.

[12] R. Watzel, K. Braun, A. Hess, W. Zuschratter, and H. Scheich, "Restoration of dendrites and spines with the objective of Topologically Correct Segmentation," in *Int. Conf. on Pattern Recognition*, pp. 472–476, 1996.

[13] W. G. Kropatsch and S. B. Yacoub, "A revision of pyramid segmentation," in *Int. Conf. on Pattern Recognition*, pp. 477–481, 1996.

[14] C. Fiorio and J. Gustedt, "Two linear time Union-Find strategies for image processing," *Theoretical Computer Science*, vol. 154, pp. 165–181, 1996.

[15] K. Mehlhorn, *Data Structures and Algorithms 1: Sorting and Searching*. Springer, 1984.

[16] M. B. Dillencourt, H. Samet, and M. Tamminen, "A general approach to connected-component labeling for arbitrary image representation," *J. of the Association for Computing Machinery*, vol. 39, pp. 253–280, April 1992.

[17] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *J. Am. Stat. Assoc.*, pp. 13–30, 1963.

6 Annex

In this part, we provide an interpretation of the segmentation and predicate criteria.
For large images, under the hypothesis that the criterion C is well chosen, The function f represents a distribution function which is a good approximation of an "ideal distribution function" f^*, that we do not know exactly. f^* satisfies the following property : the answer of $\mathcal{P}_C$ for merging two adjacent regions R and R' of I, calculated using f^* instead of f, would be *exactly* the same as the user's answer on the question "can we merge R and R'?"
This notion of exactitude is very important. It postulates that the predicate $\mathcal{P}_C$ behaves like the user when testing the merging of two regions, and therefore that C is well chosen by the user. It is obviously a very strong hypothesis difficult to ensure. But it has the property that it allows us to upperbound one type of error the algorithm can make when merging two regions. Furthermore, and we emphasize on this later in our conclusion, many parameters and knowledge can be included in the criteria C and $\mathcal{P}_C$, thus adaptatable to fit precisely to the user's behaviour.

If the previous hypothesis is not valid, there is still a partial control of the error made during the algorithm, but it is more basically a control of the error made by $\mathcal{P}_C$ with respect to C, and not with respect to the user's behaviour. From the user's point of view, it is less important, but it is relevant for the properties of our algorithm.

f^* can be seen as a similarity measure between two distinct regions of I, representing the distribution function of a random variable X. Thus, we have for example $F(x) \approx P_r(\mathrm{X} \leq x)$. Suppose we have drawn randomly two adjacent regions R, R' of I. If we make the following hypothesis:

$$\mathcal{H}_0 = \text{"}R \textit{ is different from } R'\text{"}$$

then $\mathrm{C}(R, R')$ is just a realization of X, and so a value in the domain of f^*. If the observed value $\mathrm{C}(R, R')$ satisfies

$$P_r(\mathrm{X} \leq \mathrm{C}(R, R')) \leq \alpha \qquad (5)$$

where α is fixed and small, then $\mathrm{C}(R, R')$ is small and belongs to a small set of realizations of X, having a little probability of appearance. These are the realizations of X defined by the set $\{y \leq \mathrm{C}(R, R')\}$. In that case, we can reasonably consider that R and R' in fact belong to the same region, and in that case we can merge them. That means that we reject $\mathcal{H}_0$ for the *alternative hypothesis* $\mathcal{H}_1$:

$$\mathcal{H}_1 = \text{"}R \textit{ and } R' \textit{ belong to the same region}\text{"}$$

When testing the merging of two regions R and R', there are two types of errors that can be made:

- keeping $\mathcal{H}_0$ while $\mathcal{H}_1$ is true, or
- rejecting $\mathcal{H}_0$ (*i.e.* accepting $\mathcal{H}_1$) while $\mathcal{H}_0$ is true.

These two events are noted respectively "$\mathcal{H}_0|\mathcal{H}_1$" and "$\mathcal{H}_1|\mathcal{H}_0$". It is not easy to quantify $P_r(\mathcal{H}_0|\mathcal{H}_1)$. However, provided f is a good approximation of f^*, we have:

$$P_r(\mathcal{H}_1|\mathcal{H}_0) \leq \alpha$$

Indeed, if $\mathcal{H}_0$ is true, the probability of rejecting it for $\mathcal{H}_1$ is no more than the probability of drawing two regions R and R' in I satisfying equation 5. Note that the vocabulary in the presentation of the errors is very important:

- when we choose $\mathcal{H}_0$, since it is hard to control $P_r(\mathcal{H}_0|\mathcal{H}_1)$, we say that we *keep* $\mathcal{H}_0$;
- when we choose $\mathcal{H}_1$, since we control $P_r(\mathcal{H}_1|\mathcal{H}_0)$, we can say that we *reject* $\mathcal{H}_0$ for $\mathcal{H}_1$

Thus, we obtain relatively strong bounds on one type of error, provided we choose α sufficiently small. This can be achieved by choosing α_1 in equation 2 small enough, since we also have $P_r(\mathcal{H}_1|\mathcal{H}_0) \leq \alpha_1$.