ELSEVIER

Image and Vision Computing 17 (1999) 175–187

image AND vision COMPUTING

Selectionist relaxation: genetic algorithms applied to image segmentation

Philippe Andrey*

AnimatLab, Département de Biologie, Ecole Normale Supérieure, 46 rue d'Ulm, 75230 Paris Cedex 05, France

Received 11 March 1997; received in revised form 2 March 1998; accepted 2 March 1998

Abstract

This paper describes an unsupervised image segmentation method based on a fine-grained distributed genetic algorithm. Unlike other proposed applications of genetic algorithms to this problem, the method does not require the definition of an objective fitness function evaluating candidate segmentation results. The output segmentation instead emerges as a by-product of the evolution of a population of chromosomes that are mapped onto the image and that locally adapt to its features. A sketchy analysis of the algorithm is proposed, according to which the optimal GA parameters can be predicted. The predictions are experimentally tested on artificial data. Results obtained on natural data are reported and compared with the output of a standard region segmentation method.

Keywords: Image segmentation; Genetic algorithm; Allele distribution; Optimal GA parameters

1. Introduction

Image segmentation is a low-level image processing task that aims at partitioning an image into homogeneous regions. How region homogeneity is defined depends on the application. A great number of segmentation methods are available in the literature to segment images according to various criteria such as for example gray level, color, or texture [1–3]. The output of an image segmentation algorithm can be fed as input to higher-level processing tasks, such as model-based object recognition systems.

Recently, researchers have investigated the application of genetic algorithms [4–6] to the image segmentation problem. A genetic algorithm (GA) is a stochastic search or optimization method based on the idea that natural evolution is a search process that seeks to optimize the structures it generates. In a GA, a population of candidate solutions to the problem to be solved, which are encoded into chromosomes, is iteratively updated through mechanisms of selection, crossover, and mutation. With time, good solutions spread within the population by being selected to replace bad ones, while new, possibly better, solutions are permanently being generated from previous ones through crossover and mutation.

The ability of GAs to deal with large, complex search spaces in situations where only minimum knowledge is available about the objective function, motivated their use in the context of image segmentation. For example, most existing image segmentation algorithms have many parameters that need to be adjusted. The corresponding search space is quite large and there are complex interactions among parameters. This led Bhanu et al. [7] to adopt a GA to determine the parameter set that optimize the output of a segmentation algorithm under various conditions of image acquisition. Another situation wherein GAs may be useful tools is illustrated by the work of Bhandarkar et al. [8]. In their approach to image segmentation, edge detection is cast as the problem of minimizing an objective cost function over the space of all possible edge configurations and a population of edge images is evolved using specialized operators. Results comparable with those obtained using simulated annealing are reported.

Whether the GA is used to search in the parameter space of an existing segmentation algorithm [7] or in the space of candidate segmentations [8–10], an objective fitness function, assigning a score to each segmentation, has to be specified in both cases. However, evaluating a segmentation result is itself a difficult task. To date, no standard evaluation method prevails [11] and different measures may yield distinct rankings [12]. Apart from the awkwardness of segmentation evaluation, the parameters of the GA have themselves to be adjusted. Otherwise, premature convergence towards a suboptimal solution may occur. To avoid these drawbacks, we advocate instead in favor of an unsupervised GA-based image segmentation method.

We propose an alternative approach to exploit the

* Tel: 0033 144323634; Fax: 0033 144323901.

0262-8856/99/$ - see front matter
PII: S0262-8856(98)00095-X

metaphor of natural evolution in the context of image segmentation. The image to be segmented is considered as an artificial environment wherein regions with different characteristics according to the segmentation criterion are as many ecological niches. A GA is used to evolve a population of chromosomes that are distributed all over this environment. Each chromosome belongs to one out of a number of distinct species. The GA-driven evolution leads distinct species to spread over different niches. Consequently, the distribution of the various species at the end of the run unravels the location of the homogeneous regions in the original image. Because the segmentation progressively emerges as a by-product of a relaxation process [13] mainly driven by selection, the method has been called selectionist relaxation. An attractive feature of using such a genetic algorithm is that a population of descriptors evolves on each region. Intra-region variability is thus expected to be handled efficiently. In their supervised texture segmentation technique, Jacquelin et al. [14] used a genetic algorithm to extract discriminating texture descriptors. They also pointed out the importance of using populations of descriptors to cope with textural variability.

The present work elaborates on a previous attempt [15] to exploit the metaphor described above. We have investigated several improvements to the algorithm proposed there and describe a more efficient approach in the current paper. It is characterized by the use of a real-coded GA [16], which in particular considerably alleviates the computational burden since there is no need to decode binary strings. Besides, there can be any number of region labels, which increases the robustness of the algorithm, and the labels, which are no more encoded into the chromosomes, are not subject to mutation, thus avoiding label instability. More importantly, a sketchy yet useful analysis of the new proposed algorithm is presented. The final distribution of allele values within subpopulations of chromosomes can indeed be determined. This in turn allows to predict how mutation and crossover parameters should be tuned to optimize segmentation results.

The method is generic since it can be applied to segmentation according to any criterion (gray level, texture, depth, etc.). In this paper, we focus on the problem of gray level image segmentation as an illustrative example and we discuss extensions to other criteria. The algorithm is described in Section 2. A simple model is derived and an optimality criterion is proposed. In Section 3, the validity of this analysis is tested through experimental results on synthetic data for which the desired target segmentation is exactly known. Results on various kinds of natural data are also reported. These results are confronted with the output of a classical region segmentation method.

2. Selectionist relaxation

Selectionist relaxation (SR) is an unsupervised image segmentation method, whereby the segmentation of an input image is achieved by a population of elementary units iteratively evolving through a fine-grained distributed genetic algorithm. By contrast to the standard GA scheme [4], wherein each chromosome can interact through selection and crossover with any other one in the population, interactions in distributed GAs are restricted to occur within subsets of individuals. In the coarse-grained distributed scheme [17–19], several GAs are run in parallel and periodically exchange some individuals, while in the fine-grained distributed scheme [20–22], the unique population is mapped onto a grid and interactions are restricted to occur among neighboring chromosomes. While distributed GAs have been designed primarily for the sake of efficiency, the use of a fine-grained scheme is here imposed by the two-dimensional image structure.

After having defined the chromosomes, herein referred to as units, and the fitness function, we describe the three main steps of the GA cycle (selection, crossover, and mutation). We conclude this section by an analysis of the asymptotic state of the populations, which leads to some predictions regarding the optimal tuning of the GA parameters.

2.1. Units and fitness

Segmentation though SR relies on the choice of an arbitrary feature space, wherein each image region is assumed to be represented as a cluster of points. The features that are considered depend on the objective of the segmentation. For gray level image segmentation, the feature space we have successfully experimented with is a p-dimensional space of real values in the range of the image gray level values. Each image site s is mapped in the feature space to a point $\mathbf{v}_s = (v_1,\ldots,v_p)$, which is the concatenated set of gray level values in the $p = k \times k$ image window-centered on s.

A unit is a real-coded chromosome of length p that represents a point in the feature space. The coordinates of the unit in feature space are referred to as its alleles. A population of units is built by assigning a unit to each site of the image. At site s, the fitness f_s of the associated unit $\mathbf{u}_s = (u_1,\ldots,u_p)$ is computed as the opposite of the city-block distance between $\mathbf{u}_s$ and $\mathbf{v}_s$, so that a unit whose alleles are close to the actual local pixels values will have a high fitness:

$$f_s = -\sum_{i=1}^{p} |u_i - v_i|$$

Each unit also owns a label used as a region label for the pixel where the unit is located. Initially, all units have distinct labels and their alleles are randomly generated by uniform sampling in the range of gray level values (typically, 0–255).

2.2. Relaxation cycle

The initial random population is then fed into the relaxation process, which consists in iterating the classical

```
procedure selection is
  foreach site s do
    set s* = arg max_{s'∈η_s} f_{s'}(u_{s'})
    if EDGEUNIT(u_s) then
      pick site r at random over the image
      if f_{s'}(u_r) > f_{s'}(u_{s'}), ∀s' ∈ η_s, then
        set s* = r
      endif
    endif
    replace u_s by u_{s*}
  endfor
end selection
```

Fig. 1. Pseudo-code for the selection step.

three-step GA cycle (selection, crossover, mutation). Pseudo-code for each step is given in Figs. 1–3, respectively.

2.2.1. Selection

The selection scheme implemented in SR is a local tournament selection [23]. It consists in replacing the unit located at site s by the unit having the highest fitness value in the neighborhood. The neighborhood η_s, at site s is arbitrarily defined as the 3×3 set of units centered on s. To avoid potential side-effects of sequential replacement, local tournaments take place simultaneously at all sites.

As a result of local selections, a unit can be surrounded by neighboring units all having the same label as its own label. Such a unit will be referred to as a NONEDGEUNIT. Alternatively, a unit can have at least one neighbor with a different label and will then be referred to as an EDGEUNIT. Our selection scheme departs from other local tournament selection implemented in fine grained GAs [22], since at a site occupied by an EDGEUNIT, the tournament involves an additional unit picked at random in the population. This modification allows units to propagate over regions sharing identical gray level characteristics though not being connected (a pattern that could for example results from an object being occluded). Without this mechanism, such regions would be assigned distinct labels. Besides, this modification also results in a faster propagation of the units over the image than when they are copied from one site to the next only.

2.2.2. Crossover and mutation

Crossover and mutation are applied to a unit if and only if it is a NONEDGEUNIT. A unit that fulfils this mating restriction is allowed to recombine with one of its eight neighbors, which is picked at random. Crossover is generalized uniform crossover [24,25]. For each position along the two chromosomes, an exchange occurs with probability χ, referred to as crossover rate.

```
procedure crossover is
  foreach site s do
    if NONEDGEUNIT(u_s) then
      s' = random(η_s)
      foreach i ∈ 1..p do
        if random([0; 1]) < χ then
          swap(u_{s,i}, u_{s',i})
        endif
      endfor
    endif
  endfor
end crossover
```

Fig. 2. Pseudo-code for the crossover step.

```
procedure mutation is
  foreach site s do
    if NONEDGEUNIT(u_s) then
      foreach i ∈ 1..p do
        if random([0; 1]) < μ then
          u_{s,i} ← u_{s,i} + N(0, mσ_s^2)
        endif
      endfor
    endif
  endfor
end mutation
```

Fig. 3. Pseudo-code for the mutation step.

The mutation of a unit depends on the local properties of the image at the site where the unit is located. The mutation rate μ is the probability that a position along the chromosome undergoes a mutation. Mutating the ith allele u_i of the unit located at site s consists of replacing u_i by $u_i + \delta$, where δ is randomly sampled from the Gaussian distribution $\mathcal{N}(0, m\sigma_s^2)$. m is referred to as mutation amplitude, while σ_s^2 is the variance of the local gray level distribution around site s. It is computed over the $k \times k$ image window centered on s.

2.3. Analysis

The algorithm described above departs from the conventional GA scheme [4]. In a conventional GA, the population contains candidate solutions to the problem at hand, and the fitness function scores these solutions according to their ability to solve the problem. SR contrasts with this, and with other GA-based segmentation methods [7–10], since a chromosome in the population does not code for a solution to the segmentation problem. Instead, the population itself represents a candidate segmentation, and is the unique solution that is considered each time step. The final segmentation progressively emerges as a by-product of the competition that develops among the units in the population on the basis of their individual, local fitnesses.

Accordingly, fitness does not provide global feedback evaluation of the segmentation that the population represents. This releases from the need of an objective segmentation evaluation measure, a reputed cumbersome problem. Moreover, this departure of SR from the conventional GA scheme allows for a simple analysis of final populations, as shown in the remainder of this section.

Since labels are initially unique and since crossover is only allowed to occur between units with the same label, the set of units with an identical label that have spread over a region r during the relaxation process are all issued, through successive selections, crossovers, and mutations, from the unique ancestor unit owning that label in the initial population. Let the random variable U_i denote the allele at position

i in such a set of units and R_i be the marginal law, along the ith dimension of the feature space, of the data over region r. By construction of the feature space, all R_is, $i = 1,\ldots,p$, are identically distributed. The distribution of gray levels over region r, which is assumed to be stationary Gaussian with variance σ_r^2, is the common distribution of the R_is.

The distribution of U_i can be easily determined as follows. Suppose first that crossover is switched off and let u_i^0 be the ith allele of the ancestor unit. If τ denotes the total number of generations in a run, the expected number of mutations that have accumulated at position i on a unit is $\mu\tau$. Each of these mutations added a value sampled from the centered Gaussian law with variance $m\sigma_r^2$ (σ_s^2 can be replaced by σ_r^2 because of the stationarity assumption). Since successive mutations are independent, U_i is also Gaussian and we have

$$U_i \sim \mathcal{N}(u_i^0, \mu\tau m\sigma_r^2)$$

The result remains the same if crossover is allowed. Crossover does not affect the distribution of alleles in the set of units since it only consists in exchanging alleles without modifying them.

The distribution of U_i, thus, depends on the initial allele value u_i^0 which varies from one run to the next. We propose that crossover and mutation parameters should be tuned so as to maximize the expected overlap between the two distributions R_i and U_i. Hence, they should be set so that U_i has the same variance as R_i, which leads to the following optimality condition

$$\mu m\tau = 1$$

It is then expected that the optimal mutation rate and the optimal mutation amplitude are inversely proportional. The respective values of μ and m are in fact unimportant, provided their product is optimally related to the number of generations τ. This can be interpreted as if there exists an optimal amount of mutation to be distributed over the generations, either through small but frequent, or large but rare, mutation events. Besides, one disconcerting prediction is that, as far as the final segmentation result is concerned, no benefit is to be expected from crossover.

Of course, some simplifications were made in this simple model. First, the variance σ_r^2 is of course not available during the run since regions have not yet been found by the algorithm. In practice, the variance σ_r^2 of the region r to which a site s belongs is estimated by the local variance over the $k \times k$ window centered on s. It is, therefore, important that local variances provide good estimates of the true variance of the region. Second, it is also supposed that the units have never escaped from the region whereupon they are located at the end of the run, since successive mutations have the same variance σ_r^2 in the model. This is very plausible since a unit that leaves the region to which it is adapted is highly likely to be eliminated on other regions during selection. Third, though it no doubt acts upon the distribution of alleles in the population, the role of selection has not been considered in our model. Despite these simplifications, however, we shall see in the next section that the predictions are corroborated by the experimental results.

3. Experimental results

Two sets of experiments are reported in this section. First, segmentation results on synthetic data are used to assess the validity of the analysis above. An optimal parameter combination is thus determined and is retained in a second set of experiments wherein the performance of SR is assessed on real data.

3.1. Optimal parameter setting

The SR algorithm has five parameters that must be specified before the run (Table 1). Note that, unlike conventional GAs, the size of the population is not a free parameter of the algorithm, since it is automatically determined by the image size. We will distinguish two sub-sets among the parameters of Table 1.

The first sub-set contains parameters p and τ, the optimal values of which depend upon the image to be segmented. It may indeed be necessary to adapt the area p of the windows used as feature vectors to the size of the structures one wish to detect in the image: the coarser the objects to be delineated, the larger the value of p. As far as the number of generations τ is concerned, it is necessary to choose its value according to the size of the image. This is owing to the fact that units mainly propagate from one site to the next. Hence, the larger the image, the larger the regions it contains are expected to be, and the longer is the time required by the units to spread over these regions.

On the contrary, the second sub-set contains parameters μ, m and χ, the values of which are not depending on the image. In order to determine their optimal values, we could proceed empirically and test several combinations for these parameters. Fortunately, this exhaustive, costly search can be avoided. We will only check, a posteriori, the validity of the analysis presented in Section 2.3. The objective in this subsection is to determine once and for all the optimal values for μ, m and χ.

3.1.1. Evaluation method

In order to be able to quantify segmentation performance obtained under various parameter settings, a synthetic test image, the target segmentation of which is perfectly known,

Table 1
Parameters of the SR algorithm

p	Local window area
τ	Number of generations
χ	Crossover rate
μ	Mutation rate
m	Mutation amplitude

has been used (Fig. 4(a)). This 128 × 128 pixels image is made of four regions of nearly identical areas. Each region shares a common boundary with every other one. The boundaries are either linear or curved and have either sharp or rounded corners. Within each region, gray levels are independently, identically distributed. The mean and variance of the Gaussian gray level distribution differ from one region to another.

According to the definition of Pavlidis [26,27], the regions identified by a segmentation algorithm must satisfy two properties: (1) each region is homogeneous; and (2) neighboring regions cannot be merged into a larger homogeneous one. When the target segmentation is available, objective measures can be defined to quantify the two corresponding kinds of error. Two such measures are introduced below, similar in spirit to the under- and over-merging error measures proposed in Ref. [28].

The over-segmentation cost C^+ is an average measure of the extent to which a region in the target segmentation overlaps with more than one region of the candidate segmentation, a pattern that violates condition (2) above. Consider a region r in the target segmentation and a region l of the candidate segmentation and let $p_r(l)$ be the proportion of sites belonging to r that are assigned to l. If the candidate segmentation is identical to the target, then all frequencies $p_r(l)$ but one are equal to zero. By contrast, if r is segmented into a number of equally sized regions, then all $p_r(l)$ are equal. Hence, the entropy of distribution p_r is a measure of over-segmentation in region r. The total over-segmentation cost is defined as a weighted sum of individual over-segmentations. The contribution of each region r is weighted by the proportion α_r of sites belonging to r:

$$C^+ = -\sum_r \alpha_r \sum_l p_r(l)\log p_r(l)$$

The under-segmentation cost C^- is an average measure of the extent to which a region in the candidate segmentation overlaps with several regions of the target segmentation, a pattern that violates condition (1) above. Among the sites that are assigned to a region l in the candidate segmentation, we let $q_l(r)$ denote the proportion of sites that belong to region r in the target segmentation. Ideally, all $q_l(r)$ but one are equal to zero. By contrast, if l overlaps equally with all regions of the target, then all $q_l(r)$ are equal. Therefore, the entropy of the distribution q_l is a measure of the contribution of l to under-segmentation. The total under-segmentation cost C^- is defined as a weighted sum of individual under-segmentations. The contribution of each candidate region l is weighted by the proportion β_l of sites that have been attributed to l:

$$C^- = -\sum_l \beta_l \sum_r q_l(r)\log q_l(r)$$

For a candidate segmentation, we finally define a global segmentation cost as the summed under- and over-segmentation costs. The best segmentation cost is obviously zero (Fig. 4(b)). The proposed segmentation cost has more discriminative power than the conventional misclassification rate [29]. It indeed rates distinctively candidate segmentations that would score identically using the percentage of misclassified pixels (Fig. 4(c) and (d)).

Because of random population initialization, successive runs with identical parameter sets will not score exactly the same. Therefore, 50 runs were realized for each parameter set, using independent pseudo-random numbers sequences and the average segmentation cost over the 50 runs was used as a performance criterion for the corresponding set of parameters.

3.1.2. Optimal mutation

Here, we test the prediction derived from the analysis presented above, according to which the optimal product of mutation rate and mutation amplitude follows $\mu m = 1/\tau$. To achieve this goal, we determine experimentally the performance of the algorithm under various combinations of μ and m, and we check whether or not the combinations that minimize segmentation cost obey this relationship. The size of pixels matrices is $p = 3 \times 3$ and the number of generations is $\tau = 64$. No crossover is allowed, hence $\chi = 0$.

3.1.2.1. Experiment 1. The value of m being fixed, the average segmentation cost has been determined for $\mu = \hat{\mu}/8$, $\hat{\mu}/4$, $\hat{\mu}/2$, $\hat{\mu}$, $2\hat{\mu}$, $4\hat{\mu}$, and $8\hat{\mu}$, where $\hat{\mu} = 1/m\tau$. This experiment has been repeated using three distinct values for $m = 1.0$, 0.5 and 0.2.

From the obtained results (Fig. 5), it can be stated that: (1)

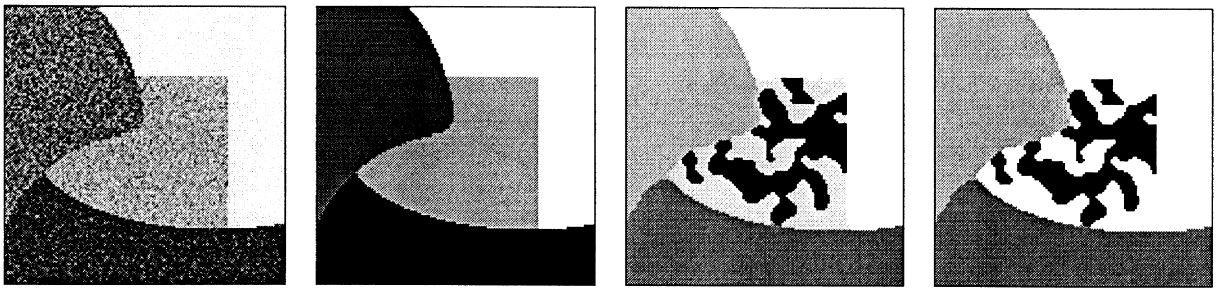

Fig. 4. Test image and segmentation costs. The three putative segmentations of test image (a) have segmentation costs equal to: (b) 0.0; (c) 0.26; and (d) 0.49.

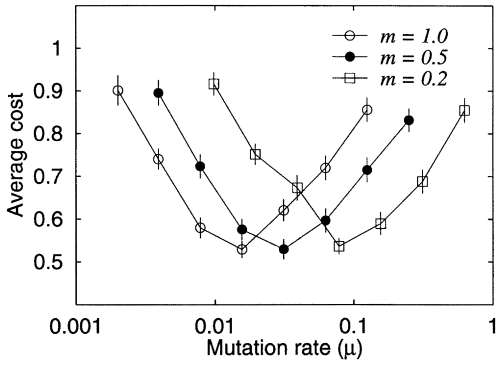

Fig. 5. Average segmentation cost as a function of mutation rate, at three values of the mutation amplitude. The height of each error-bar equals twice the standard-error of the corresponding 50-sample.

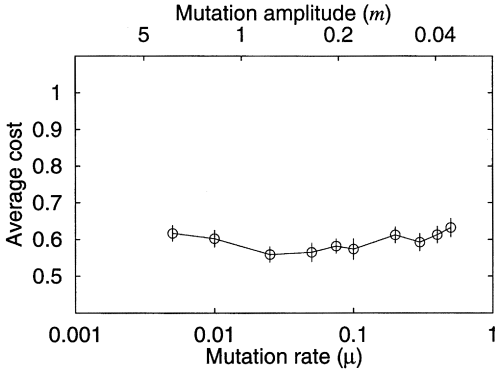

Fig. 7. Average segmentation cost for different couples of values μ, m satisfying $\mu m\tau = 1$.

for a given value of the mutation amplitude m, there exists a unique value μ^* of the mutation rate μ that minimizes the segmentation cost; (2) the value $\hat{\mu}$ derived from analysis correctly predicts the observed value μ^*; (3) mean performance depends solely upon the product μm, whatever the respective values of μ and m may be (this is deduced from the fact that, under the semi-logarithmic scale used to plot the curves, they are reciprocally translated).

3.1.2.2. Experiment 2. Reciprocally, the value of μ being fixed, the average segmentation cost has been determined for $m = \hat{m}/8$, $\hat{m}/4$, $\hat{m}/2$, $\hat{m}$, $2\hat{m}$, $4\hat{m}$, and $8\hat{m}$, where $\hat{m} = 1/\mu\tau$. The experiment has been repeated using three values for $\mu = 0.016$, 0.03 and 0.08.

The obtained results (Fig. 6) confirm that, for a given value of μ, there exists a unique value m^* minimizing segmentation cost. Moreover, this value depends upon μ and is accurately predicted by the value $\hat{m}$ derived analytically. For $m \leq \hat{m}$, the score depends solely upon the product μm, as in Experiment 1. For $m > \hat{m}$, however, significantly different performances can be observed for the same μm product but simultaneously varying μ and m values.

3.1.2.3. Experiment 3. The results of Experiments 1 and 2 suggest that, provided the product μm is at its optimal predicted value $1/\tau$, it solely determines the average observed cost, whatever the respective values of μ and m may be. We have tested the validity of this result over a larger number of values μ, m satisfying the constraint $\mu m\tau = 1$.

As expected, the performance remains remarkably constant when μ and m simultaneously vary (Fig. 7). The slope of the regression line between segmentation cost and mutation rate is not significantly different from zero ($t = 1.834$, $N = 500$). Of course, since μ and m are inversely proportional in this experiment, there is no significant binding between segmentation cost and mutation amplitude either.

However, the fact that the cost remains constant when μ and m vary simultaneously hides the fact that the obtained segmentations are not completely equivalent. Indeed, the number of labels in the final segmentations varies significantly (Fig. 8). It appears that there exists a limit value μ_l such that, for $\mu < \mu_l$, the number of labels decreases when μ increases, while it remains constant for $\mu > \mu_l$ (of course, a reciprocal statement can be made with regards to m).

By contrast with the average number of labels, the average label size, defined as the mean number of pixels to which a label is assigned, increases when mutation rate increases (Fig. 8). This increase can result from two, non necessarily exclusive phenomena. On the one hand, it can reflect the fact that the most prominent labels increase their

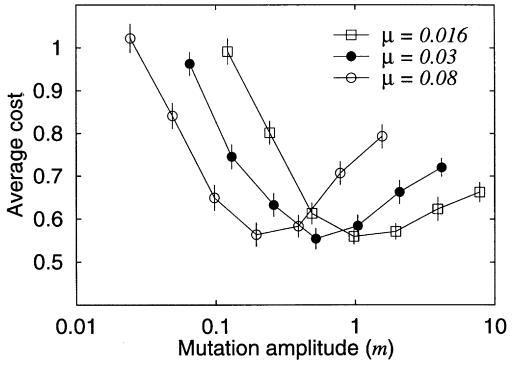

Fig. 6. Average segmentation cost as a function of mutation amplitude, at three values of the mutation rate.

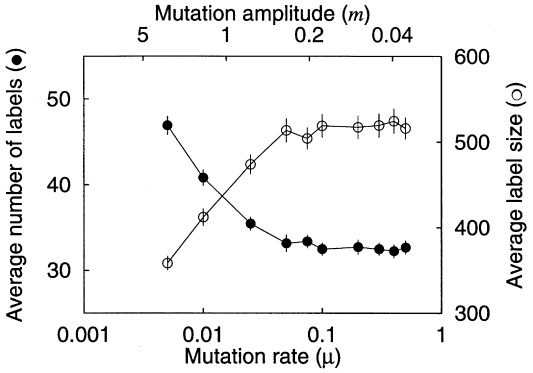

Fig. 8. Average number of labels and average label size for different couples μ, m satisfying $\mu m\tau = 1$.

size at the expense of the less represented labels, which disappear. On the other hand, the most prominent labels can remain at the same size, while their number is increased, again at the expense of the less represented labels. In order to decide between these two hypotheses, the evolution of the number of labels occupying more or less than 1% of the image have been determined (data not shown). The number of labels occupying more than 1% of the total number of pixels does not vary significantly with mutation rate ($t = 1.649$, $N = 500$). On the contrary, the number of labels occupying less than 1% of the image size diminishes significantly when the mutation rate is increased ($t = -8.648$, $N = 500$, $P < 0.001$). From these results, it can be concluded that the increased average label size is only the consequence of the elimination of the less represented labels at the benefit of the most represented ones.

3.1.2.4. Discussion (Experiments 1–3). This experimental study corroborates the analysis presented in Section 2.3. Indeed, it has been found that the product μm solely determines the observed performance, at least provided the constraint $\mu m\tau = 1$ is satisfied. The respective values of μ and m can influence performance only in cases where μm is not at its predicted optimum value.

Besides, this study provides unexpected results. Segmentations with identical average cost were obtained, that differ with respect to the number of labels, and, consequently, with respect to the average label size. Such differences are not reflected in the cost value. This can be attributed to the fact that the labels that disappear when mutation rate increases are the less represented ones. Because of the definition of C^+ and C^-, these are precisely the labels that only weakly contribute to the cost value.

3.1.2.5. Conclusion. As far as segmentation cost is concerned, the number of optimal settings for mutation rate and amplitude is infinite. However, a smaller number of labels is also desirable because it is closer to the ideal number of labels. Therefore, μ (resp. m) should be taken large (resp. small) enough so as to minimize the number of labels. There is, however, no benefit in choosing μ too large because this increases the computation time while providing no improvement beyond a certain μ value. This is why we have chosen to arbitrarily set mutation amplitude at $m = 1$, which automatically leads to set mutation rate at $\mu = 1/\tau$.

3.1.3. Optimal crossover

In the previously reported experiments, no recombination between units was allowed to occur. In the following experiments, the validity of the analysis of Section 2.3, according to which crossover rate should not influence the observed segmentation performance, is tested.

3.1.3.1. Experiment 4. The average segmentation cost has been recorded for various crossover rates. Mutation rate and amplitude are set to $\mu = 1/\tau$ and $m = 1$. As above, $p = 3 \times 3$ and $\tau = 64$.

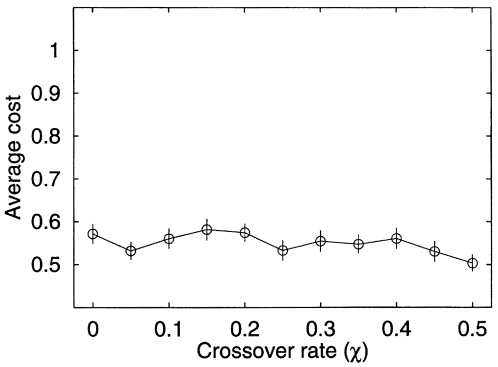

Fig. 9. Average segmentation cost as a function of crossover rate.

The observed performance (Fig. 9) remains remarkably constant over the range of tested χ values. Despite a slight overall decreasing tendency, the slope of the regression line between segmentation cost and crossover rate is not significantly different from zero ($t = -1.828$, $N = 550$). As far as the final segmentation cost is concerned, it thus appears that crossover does not yield any improvement. On the contrary, crossover appears as a burdensome operator since it increases the computational needs without increasing performance.

However, crossover significantly affects the number and the size of the labels (Fig. 10). The greater the crossover rate, the smaller the number of labels, and, conversely, the larger their size. As in Experiment 3, the hypothesis that the number of large labels increases with crossover rate has been tested by determining the number of labels that occupy more or less than 1% of the image size (data not shown). It has to be rejected since the number of labels occupying more than 1% of the image does not significantly vary with crossover rate ($t = -0.030$, $N = 550$). On the contrary, the number of labels that represent less than 1% of the image significantly decreases as crossover rate is increased ($t = -9.568$, $N = 550$, $P < 0.001$).

Though it does not affect the final segmentation cost, crossover may speed up convergence of the cost towards

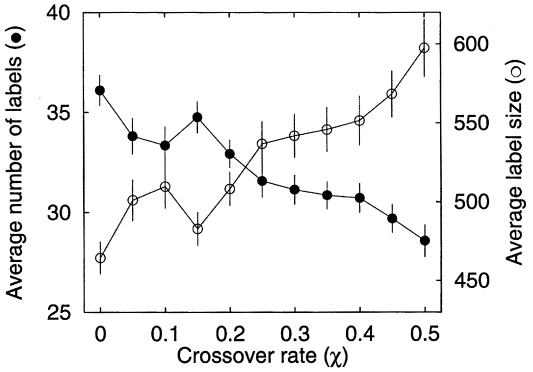

Fig. 10. Average number of labels and average label size as a function of crossover rate.

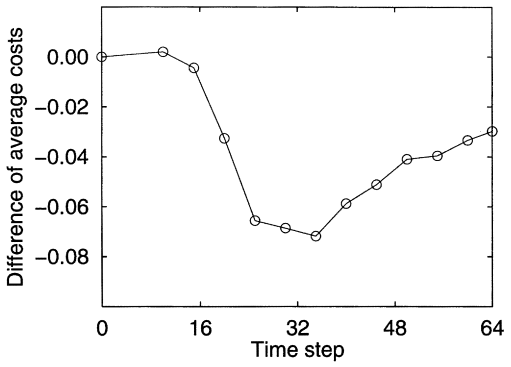

Fig. 11. Difference between average costs in two conditions ($\chi = 0.5$ and $\chi = 0.0$) as a function of time step.

its final value. To test this hypothesis, the evolution of the cost during generations has been recorded in the two extreme cases $\chi = 0$ and $\chi = 0.5$. The evolution with time of the difference between average costs is plotted in Fig. 11. While segmentation costs in both conditions are identical or close to each other at the beginning and at the end of the runs, they do not evolve at the same rate. Convergence is faster when $\chi = 0.5$. Similar results were obtained with $\chi = 0.1$ and $\chi = 0.25$ (data not shown), while the amplitude of the differences were less pronounced. These results illustrate that the cost of the intermediary segmentations diminishes when crossover rate is increased.

3.1.3.2. Discussion. The results of the previous experiment do not demonstrate a significant influence of crossover rate over segmentation cost, when μ and m are set to their optimal values. This is consistent with the analysis presented in Section 2.3. Indeed, crossover does not alter the distribution of the alleles in a sub-population of units along any dimension of the feature space since it only exchanges alleles within this sub-population.

However, crossover rate significantly influences the number of labels, and, consequently, their size. This effect has no counterpart on the observed segmentation cost because the labels that disappear when crossover rate is increased are the less represented ones, which only weakly contribute to the cost value.

Besides, it was demonstrated that convergence is affected by crossover. During the run, a sequence of segmentations is generated, which constitutes a trajectory in the space of possible segmentations. The results illustrate that the cumulated cost along the trajectory diminishes when crossover rate is increased. A plausible interpretation of this result is that, when crossover rate is high, good combinations of alleles appear early during the evolution as a result of intense recombination. The final cost is, however, not statistically different when crossover is disabled because the total amount of mutation was optimally tuned.

3.1.3.3. Conclusion. The choice of an optimal crossover rate value cannot be based on segmentation cost because crossover has no influence on it. However, since segmentations with fewer labels are closer to the ideal one, crossover should be chosen as large as possible, namely $\chi = 0.5$. Another reason for choosing χ large is that it speeds up convergence.

3.1.4. Independence between crossover and mutation

3.1.4.1. Experiment 5. Based on analysis, it has been assumed that the optimal values for μ and m were independent of crossover rate. To check the validity of this assumption, Experiments 1 and 2 have been repeated using another χ value. The same protocol has been adopted, but crossover rate is now at its optimal value $\chi = 0.5$. The average segmentation cost has been determined as a function of μ, keeping $m = 0.5$ constant, and as a function of m, keeping $\mu = 0.03$ constant.

The obtained results (Fig. 12) confirm that the values of μ and m that minimize the segmentation cost are independent of χ. Indeed, when μ only is varied, the best average costs when $\chi = 0.0$ and when $\chi = 0.5$ are observed for the same value μ^*. A similar result is obtained when m alone is varied. However, the results also suggest an influence of crossover on performance when μ and m are not optimal. Indeed, for $\mu < \mu^*$, the cost is lower for $\chi = 0.5$ than for $\chi = 0.0$, while the converse is observed for $\mu > \mu^*$. Similar statements hold for the mutation amplitude m. However, when μ and m are optimal, it is observed that the

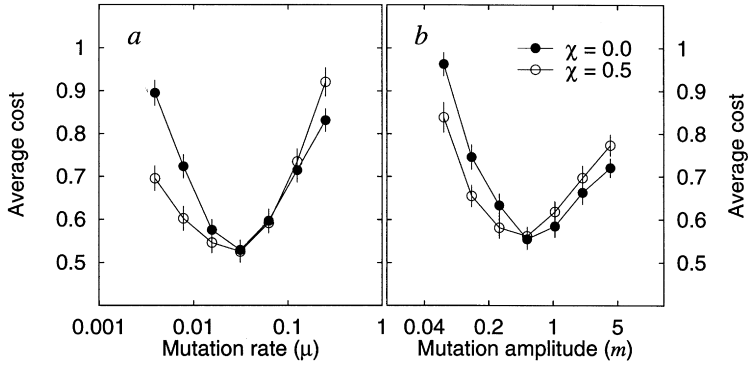

Fig. 12. Optimal mutation rate and amplitude are independent of crossover rate.

segmentation cost does not depend on crossover rate, which is consistent with the results of Experiment 4.

3.1.4.2. Experiment 6. Reciprocally, we assumed that the optimal χ value determined in Experiment 4 was independent of the respective values of μ and m, provided μm was optimal. The validity of this assumption has been tested by repeating Experiment 4 in two new conditions, $\mu = 1/p$ and $m = p/\tau$, on the one hand, and $\mu = 1/2p$ and $m = 2p/\tau$, on the other hand.

The obtained results (data not shown) are similar to those of Experiment 4: the average segmentation cost is independent of crossover rate; the number of labels decreases, and their size increases, when crossover rate increases; convergence is faster with larger χ values.

3.1.4.3. Conclusion. The above results corroborate the independence between optimal mutation rate and amplitude, on the one hand, and crossover rate, on the other hand. Conversely, the optimal crossover rate value is independent of mutation rate and amplitude as long as the product of these two parameters is optimal.

3.2. Segmentation of natural scenes

The optimal parameter set determined in the previous experimental study on artificial data has been retained for segmenting natural data. SR has been tested on a number of real images, including outdoor, indoor, and biomedical scenes [30]. A representative subset of results is reported in this section. All images are coded on 256 gray levels and have the same size of 256×256 pixels. No pre- nor post-processing is applied to the images.

The size $p = k \times k$ of the image windows used as feature vectors is the only parameter for which a rough tuning has been attempted for each image. In each case, two values $k = 3$ and $k = 5$ were tested and the value yielding the visually most consistent result was selected (Table 2). All other parameters remain constant throughout experiments. Crossover and mutation parameters are set to their optimal values as predicted by the analysis complemented with the experimental studies reported above, that is to say, $\chi = 0.5$, $\mu = 1/\tau$ and $m = 1$. As already mentioned, the number of generations τ must be selected according to the size of the image to allow sufficient time for the units to spread over regions. For 256×256 pixels images, we observed that the spreading was generally completed within $\tau = 150$ generations. In fact, it may be necessary to choose τ according to the desired degree of stability of the final segmentation. Instead of setting an arbitrary stability threshold at which the segmentation should stop, each image is segmented through five independent runs of 150 generations each and the most stable one is reported.

Table 2
Segmentation of natural scenes: SR and S&M parameter values

Image	SR	S&M
Fig. 13(a)	3×3	21
Fig. 14(a)	5×5	30
Fig. 15(a)	3×3	13

SR results are compared with the results obtained using a split-and-merge (S&M) region segmentation method [26]. This method consists in first recursively splitting and merging an initial partition of the image into square regions. A grouping procedure is subsequently applied to merge regions that could not be merged during the first stage because of the quad-tree representation of the image. We have attempted to optimize the output of this algorithm on each image. In the original version of the algorithm [26], the region uniformity criterion according to which merging, splitting, and grouping decisions are made is based on the difference between extreme gray level values. For example, two regions are grouped if the difference between maximum and minimum gray levels in the resulting region does not exceed a user-specified threshold. Because of the extremely poor results obtained with this procedure, we have instead used an homogeneity criterion based on the absolute difference of mean gray level values. Hence, two regions are grouped if the difference between their mean gray level values does not exceed a threshold. The threshold value was manually adjusted on each image to obtain the visually most consistent segmentation (Table 2). The threshold was selected low enough so that pertinent structures were correctly delineated, while at the same time high enough so that unnecessary region over-splitting was prevented from occurring.

The results obtained on an outdoor scene are shown in Fig. 13. We have no quantitative criterion to assess the validity of this segmentation (in Section 3.1, such a criterion was available because a synthetic test image, with a known region map, was used). However, the result is good in the sense that regions with visually distinct mean gray level values are assigned distinct labels. Fig. 14 illustrates the segmentation of the image of a breast tumor histological slice. Again, a visually and structurally consistent result is obtained. Besides, it appears that, though larger windows were used in this case as compared with the previous one, the position of the boundaries identified by the algorithm is as accurate as previously. It thus appears that increasing k does not degrades the accuracy of region boundaries localization, which contrasts with what could have been expected. Indeed, performance of image segmentation methods is often hampered by the existing balance between boundary detection accuracy (which increases with window or filter size) and edge localization [31].

The comparison between SR and S&M results suggests several points. It is first noted that in both cases, some regions that would be segmented by a human subject are in fact merged. This is, for example, the case between grass and bricks on the left side of Fig. 13(c) and between bush and bricks on the left side of Fig. 13(d). Such patterns are

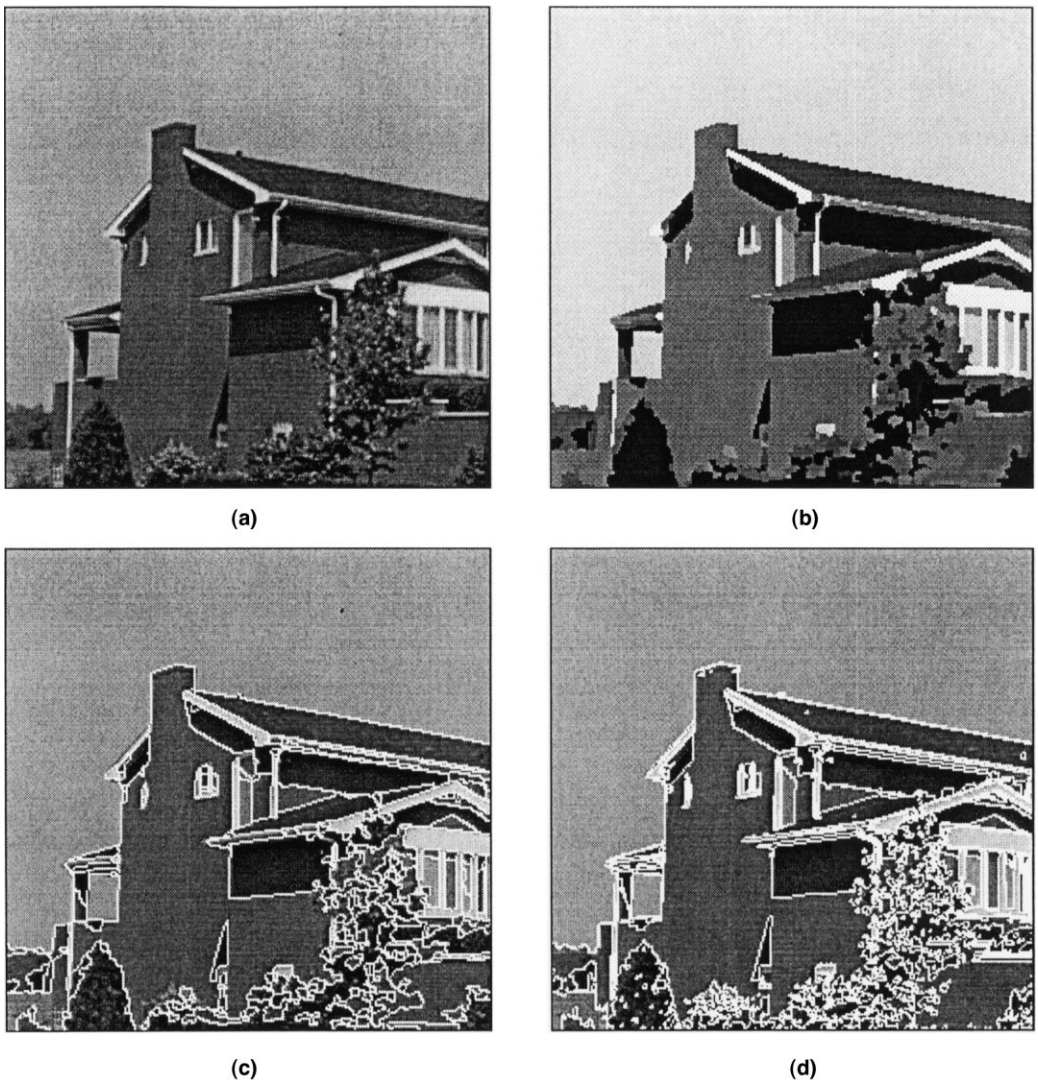

Fig. 13. Segmentation of an outdoor image: (a) input image; (b) SR labels; (c) SR mapped contours; (d) S&M mapped contours.

not unexpected if gray level information alone is used to generate the segmentation. Other information such as textural features (e.g. Ref. [32]) should be incorporated to separate such regions. Another aspect illustrated by these results is that it is very difficult, if not impossible, to find a unique satisfactory global parameter tuning when different regions of the image call for conflicting parameter settings. Indeed, the most satisfactory S&M segmentation result erroneously merges roof and bricks (see the bottom of the rain-pipe in the middle of Fig. 13(d)) despite the quite distinct gray level values between these regions. This points towards the advantage of using a spatially adaptive segmentation technique. This is precisely what SR offers, a property that is further illustrated on Fig. 14. The comparison between SR and S&M results shows that S&M ability at identifying a region as a unique entity degrades rapidly as the variance of gray levels increases. The two methods indeed locate correctly the lumen of the canals, over which gray level variance is low, while only SR accurately segments the conjunctive tissue, over which gray level variance is much higher. To avoid over-segmentation of the conjunctive tissue through S&M, a higher threshold should be used, which would results in merging the lumen and the epithelia of the canals. These results outline a major feature of SR, wherein populations of units can adapt locally to each region, thanks to the adaptive, site-dependent mutation scheme.

Consistent with this pattern of results is the fact that, on an image wherein almost all regions are smooth, such as the indoor scene of Fig. 15, the differences between the two methods are less pronounced. The SR result in this case is not as satisfactory as the previous ones. This can be explained by the fact that the image contains almost exclusively smooth surfaces with a gradient of illumination. Such regions depart from the stationary Gaussian region model assumed in Section 2.3. To deal with such a category of image data, the mutation operator and the analysis should be modified in accordance with the distribution of the data.

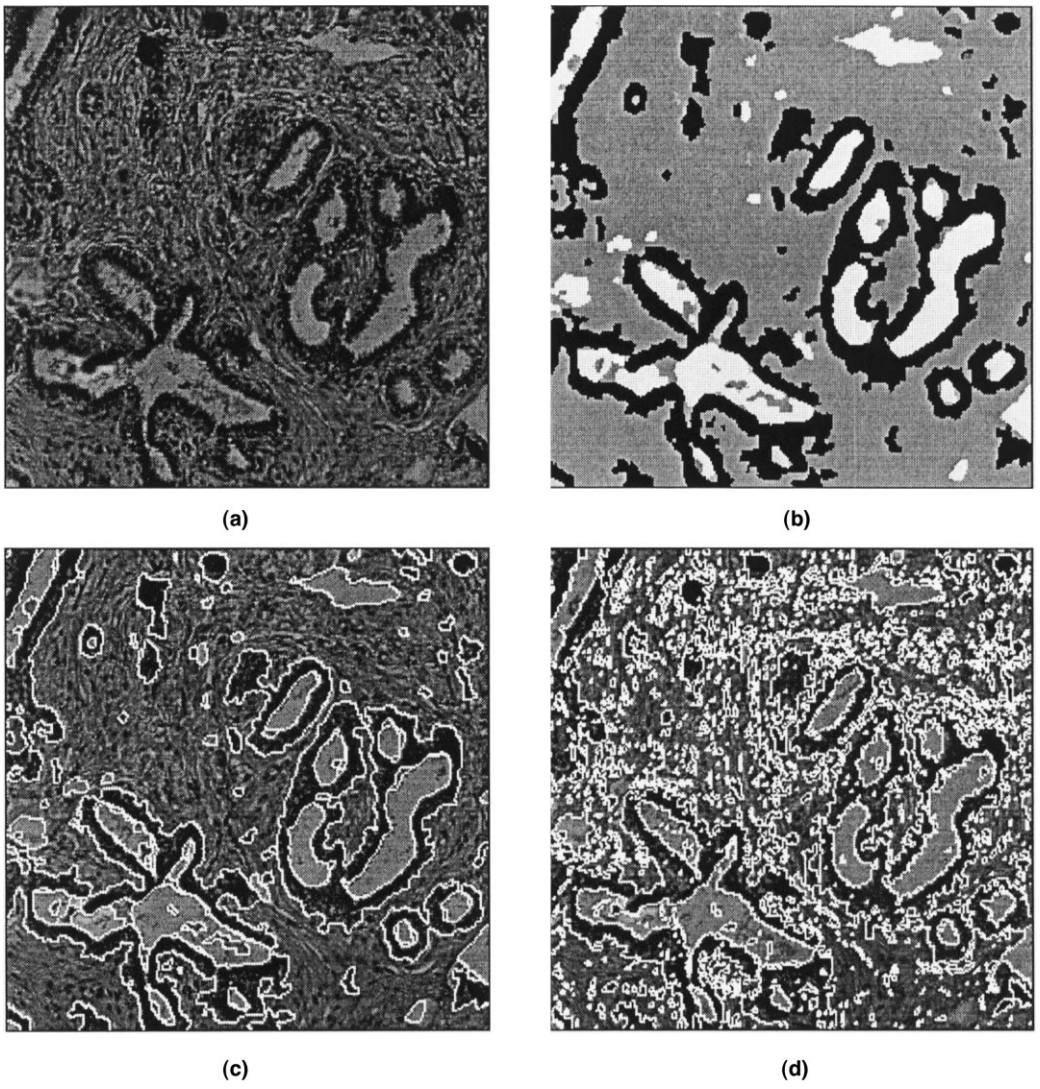

Fig. 14. Segmentation of an histological slice image: (a) input image; (b) SR labels; (c) SR mapped contours; (d) S&M mapped contours.

Since SR is a relaxation method that iterates over generations, and because the size of the population in the GA is quite large (it is equal to the size of the image), computing time of SR is of course, greater than with S&M. On a Sun SPARCstation-20, 150 generations for a 256×256 pixels image are typically completed within 3 min, while it requires around 1 s to segment the same image through S&M.

4. Final discussion and conclusion

We have described an unsupervised image segmentation method based on a fine-grained distributed GA. Two main points characterize the departure of SR from other applications of GAs to the problem of image segmentation [7–10].

Firstly, no segmentation evaluation criterion is required in SR. The segmentation cost of Section 3.1 was introduced only to evaluate, a posteriori, the segmentation results obtained under various experimental conditions designed to assess the validity of the analysis. Among the available segmentation performance measures [11], other criteria could have been used as well. The conclusions of Section 3.1 are in particular unchanged if the percentage of misclassified pixels [29] is used instead of the above mentioned criteria (data not shown).

Secondly, some of the behavior of SR can be captured in a simple model. This in turn allows to analytically determine the optimal parameter combination. Experimental results corroborating this analysis were reported. Parameter tuning in other GA-based segmentation techniques cannot rely on the current state of the GA theory. The theoretical analysis of GAs is still an open research area [6].

Despite the available guidelines to set mutation and cross-over parameters, SR is not a parameter-free algorithm since the number of generations and the area of pixel windows used as feature vectors must be specified externally.

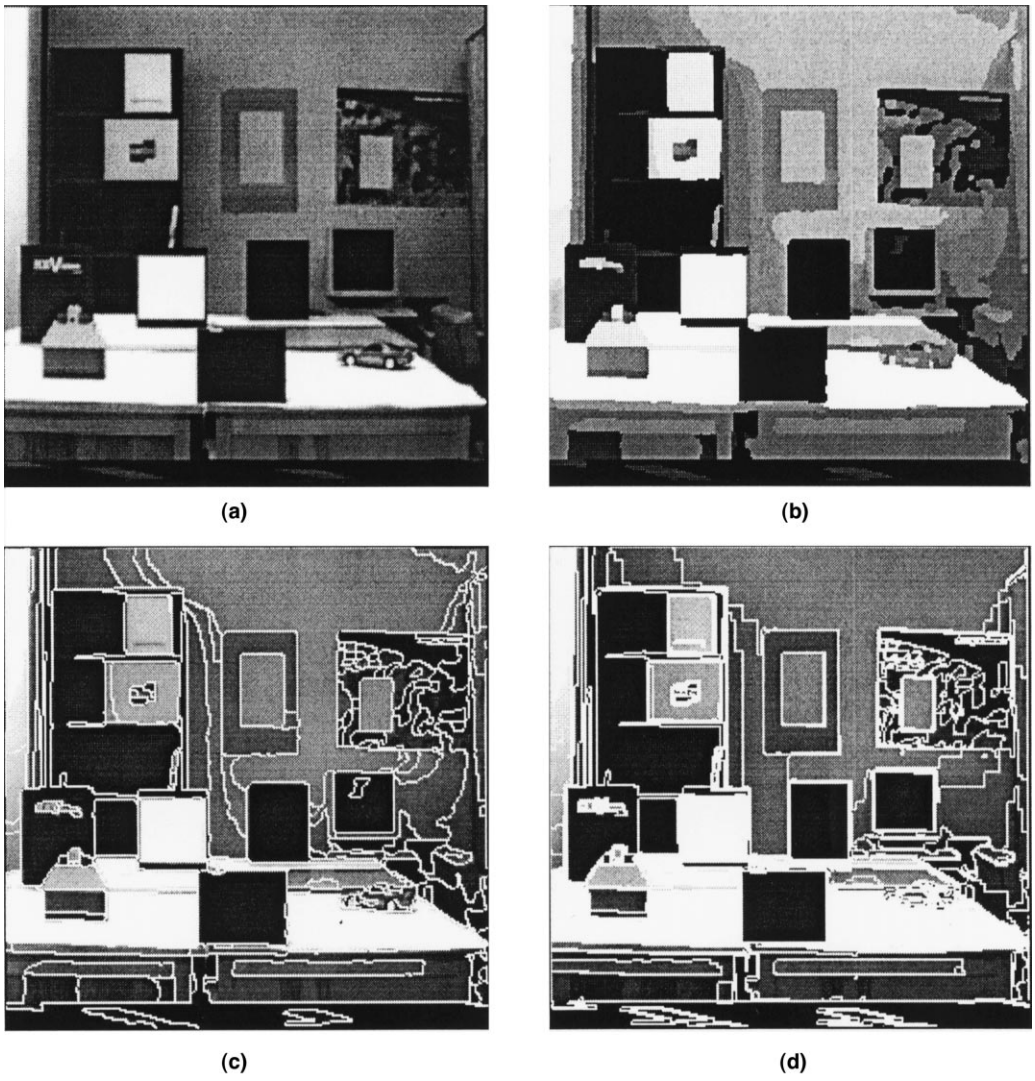

Fig. 15. Segmentation of an indoor scene: (a) input image; (b) SR labels; (c) SR mapped contours; (d) S&M mapped contours.

However, their tuning is facilitated by the fact that the former is indexed on the size of the image and only a few choices (here, no more than two) have to be considered for the latter. In contrast, by trying to overcome the difficulty of parameter tuning of existing segmentation algorithms, conventional applications of GAs to the segmentation problem introduce the difficulty of tuning the parameters of the GA itself. Besides, the conventional approach requires an objective fitness function, which may itself depend on parameters that have to be heuristically tuned.

The work presented herein can be pursued in two main directions. It is first of interest to study how the current scheme can be applied to segmentation according to other criteria than gray level. A first attempt towards textured image segmentation is reported in Ref. [33], but there, the parameters of the GA were empirically tuned. Generalizing the operators and the current analysis to other feature spaces wherein distinct features are represented along different dimensions is straightforward. The second point of interest is to extend the analysis to the dynamics of the allele distributions towards their asymptotic state. Despite its simplicity, the distribution of the alleles has indeed proven to be a useful tool. Using this concept to capture the dynamics of SR may suggest improved operators and parameter tuning, thus possibly resulting in increased convergence speed and final segmentation quality.

References

[1] R.M. Haralick, L.G. Shapiro, Survey: image segmentation techniques, Computer Vision, Graphics and Image Processing 29 (1985) 100–132.
[2] N.R. Pal, S.K. Pal, A review on image segmentation techniques, Pattern Recognition 26 (9) (1993) 1277–1294.
[3] T.R. Reed, H.J.M. du Buf, A review of recent texture segmentation and feature extraction techniques, CVGIP: Image Understanding 57 (3) (1993) 359–372.
[4] D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison Wesley, Reading, MA, 1989.

[5] J.H. Holland, Adaptation in Natural and Artifical Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence, The MIT Press/A Bradford Book, Cambridge, MA, 1992. 1st ed.: University of Michigan Press, Ann Arbor, MI, 1975.
[6] S. Forrest, Genetic algorithms: principles of natural selection applied to computation, Science 261 (1993) 872–878.
[7] B. Bhanu, S. Lee, J. Ming, Adaptive image segmentation using a genetic algorithm, IEEE Transactions on Systems, Man, and Cybernetics 25 (12) (1995) 1543–1567.
[8] S.M. Bhandarkar, Y. Zhang, W.D. Potter, An edge detection technique using genetic algorithm-based optimization, Pattern Recognition 27 (9) (1994) 1159–1180.
[9] G. Seetharaman, O.S. Prabhu, A. Narasimhan, Two modified crossover and mutation operators for image segmentation by genetic algorithms, in: S.-S. Chen (Ed.), Neural and Stochastic Methods in Image and Signal Processing, vol. 1766 of SPIE Proceedings, 1992, pp. 66–76.
[10] D.N. Chun, H.S. Yang, Robust image segmentation using genetic algorithm with a fuzzy measure, Pattern Recognition 29 (7) (1996) 1195–1211.
[11] Y.J. Zhang, A survey on evaluation methods for image segmentation, Pattern Recognition 29 (8) (1996) 1335–1346.
[12] Y.J. Zhang, J.J. Gerbrands, Objective and quantitative segmentation evaluation and comparison, Signal Processing 39 (1994) 43–54.
[13] L.S. Davis, A. Rosenfeld, Cooperating processes for low-level vision: a survey, Artificial Intelligence 17 (1981) 245–263.
[14] C. Jacquelin, A. Aurengo, G. Hejblum, Evolving descriptors for texture segmentation, Pattern Recognition 30 (7) (1997) 1069–1079.
[15] P. Andrey, P. Tarroux, Unsupervised image segmentation using a distributed genetic algorithm, Pattern Recognition 27 (5) (1994) 659–673.
[16] A.H. Wright, Genetic algorithms for real parameter optimization, in: G.J. Rawlins (Ed.), Foundations of Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1991, pp. 205–218.
[17] C.B. Pettey, M.R. Leuze, J.J. Grefenstette, A parallel genetic algorithm, in: J.J. Greffenstette (Ed.), Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms, Lawrence Erlbaum Associates, Hillsdale, NJ, 1987, pp. 155–161.
[18] R. Tanese, Distributed genetic algorithms, in: J.D. Schaffer (Ed.), Proceedings of the Third Intemational Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1989, pp. 434–439.
[19] J.P. Cohoon, S.U. Hedge, W.N. Martin, D.S. Richards, Distributed genetic algorithms for the floorplan design problem, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 10 (4) (1991) 483–492.
[20] B. Manderick, P. Spiessens, Fine-grained parallel genetic algorithms, in: J.D. Schaffer (Ed.), Proceedings of the Third Intemational Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1989, pp. 428–433.
[21] H. Mühlenbein, Parallel genetic algorithms, population genetics and combinatorial optimization, in: J.D. Schaffer (Ed.), Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1989, pp. 416–421.
[22] P. Spiessens, B. Manderick, A massively parallel genetic algorithm. Implementation and first analysis, in: R.K. Belew, L.B. Booker (Eds), Proceedings of the Fourth International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1991, pp. 279–286.
[23] D.E. Goldberg, K. Deb, A comparative analysis of selection schemes used in genetic algorithms, in: G.J. Rawlins (Ed.), Foundations of Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1991, pp. 69–93.
[24] G. Syswerda, Uniform crossover in genetic algorithms, in: J.D. Schaffer (Ed.), Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1989, pp. 2–9.
[25] K.A. De Jong, W.M. Spears, A formal analysis of the role of multipoint crossover in genetic algorithms, Annals of Mathematics and Artificial Intelligence 5 (1) (1992) 1–26.
[26] S.L. Horowitz, T. Pavlidis, Picture segmentation by a tree traversal algorithm, Journal of the Association for Computing Machinery 23 (2) (1976) 368–388.
[27] T. Pavlidis, Structural Pattern Recognition, Springer, New York, 1977.
[28] M.D. Levine, A. Nazif, An experimental rule-based system for testing low level segmentation strategies, in: K. Preston, Jr, L. Uhr (Eds), Multicomputers and Image Processing. Algorithms and Programs, Academic Press, New York, 1982, pp. 149–160.
[29] W.A. Yasnoff, J.K. Mui, J.W. Bacus, Error measures for scene segmentation, Pattern Recognition 9 (1977) 217–231.
[30] R. Andrey, Segmentation d'images par algorithmes genetiques, Thèse de l'Université Paris 7 Spècialitè Biomathèmatiques, 1997.
[31] J. Canny, A computational approach to edge detection, IEEE Transactions on Pattern Analysis and Machine Intelligence 8 (6) (1986) 679–698.
[32] R.M. Haralick, K. Shanmugam, I. Dinstein, Textural features for image classification, IEEE Transactions on Systems, Man, and Cybernetics 3 (6) (1973) 610–621.
[33] P. Andrey, P. Tarroux, Unsupervised segmentation of Markov random field modeled textured images using selectionist relaxation, IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (3) (1998) 252–262.