



A Cognitive Vision Approach to Image Segmentation

Vincent Martin, Monique Thonnat

► To cite this version:

Vincent Martin, Monique Thonnat. A Cognitive Vision Approach to Image Segmentation. Paula Fritzsche. Tools in Artificial Intelligence, InTech Education and Publishing, pp.265-294, 2008, 978-953-7619-03-9. inria-00499604

HAL Id: inria-00499604

<https://inria.hal.science/inria-00499604>

Submitted on 10 Jul 2010

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

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

A Cognitive Vision Approach to Image Segmentation

Vincent Martin and Monique Thonnat
INRIA Sophia Antipolis Méditerranée, project-team PULSAR
France

1. Introduction

Image segmentation consists in grouping pixels sharing some common characteristics. In vision systems, the segmentation layer typically precedes the semantic analysis of an image. Thus, to be useful for higher-level tasks, segmentation must be adapted to the goal, i.e. able to effectively segment objects of interest. Our objective is to propose a cognitive vision approach to the image and video segmentation problem. More precisely, we aim at introducing learning and adaptability capacities into the segmentation task. Traditionally, explicit knowledge is used to set up this task in vision systems. This knowledge is mainly composed of image processing programs (e.g., specialized segmentation algorithms and post-processing's) and of program usage knowledge to control segmentation (e.g., algorithm selection and algorithm parameter settings).

In real world applications, when the context changes, so does the appearance of the images. It can be due to local changes (e.g., shadows, reflections) and/or global illumination changes (e.g., due to meteorological conditions). The consequences on segmentation results can be dramatic. This context adaptation issue emphasizes the need of automatic adaptation capabilities. Our first objective is to learn the contextual variations of images in order to discriminate between different segmentation actions. The identification of the contexts will lead to different segmentation actions as algorithm selection.

When designing a segmentation algorithm, internal parameters (e.g., thresholds or minimal sizes of regions) are set with default values by the algorithm authors. In practice, it is often up to an image processing expert to supervise the tuning of these free parameters to get meaningful results. As seen in Figure 1, it is not clear how to choose the best parameter set regarding the segmented images: the first one is quite good but several parts of the insect are missing; the second one is also good, since the insect is well outlined, but too many meaningless regions are also present. However, complex interactions between free parameters make the behaviour of the algorithm fairly impossible to predict. Moreover, this awkward task is tedious and time-consuming. Thus, the algorithm parameter tuning is a real challenge. To solve this issue, our objective is threefold: first, we want to automate this task in order to alleviate users' effort and prevent subjective results. Second, the fitness function used to assess segmentation quality should be generic (i.e. not application dependent). Third, no *a priori* knowledge of segmentation algorithm behaviours is required, only ground truth data should be provided by users.

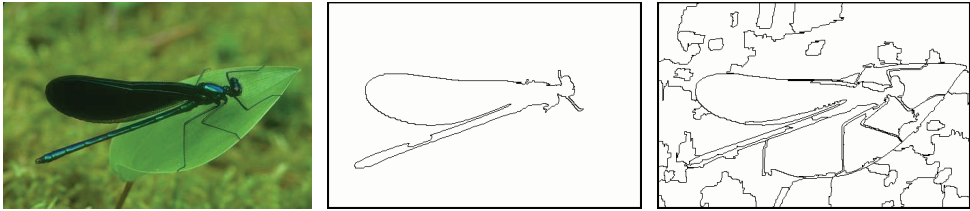


Fig. 1. Illustration of the problem of algorithm parameter tuning. An image is segmented with the same algorithm (based on colour homogeneity) tuned with two different parameter sets.

The very first problem of segmentation is that a unique general method still does not exist: depending on the application, algorithm performances vary. This is illustrated in Figure 2 where two different algorithms are applied on the same image. The first one seems to be visually more efficient to separate the ladybird from the leaf. The second one produces too many regions not very meaningful. Basically, two popular approaches exist to set up the image segmentation task in a vision system. A first approach is to develop a new segmentation algorithm dedicated to the application task. A second approach is to empirically choose an existing algorithm, for instance by a trial-and-error procedure. The first approach leads to develop an *ad hoc* algorithm, from scratch, and for each new application. The second approach does not guarantee adapted results and robustness. So, a need exists for developing a new approach to the algorithm selection issue. When facing different algorithms, this approach should be able to automatically choose the one best suited with a segmentation goal and the image content.

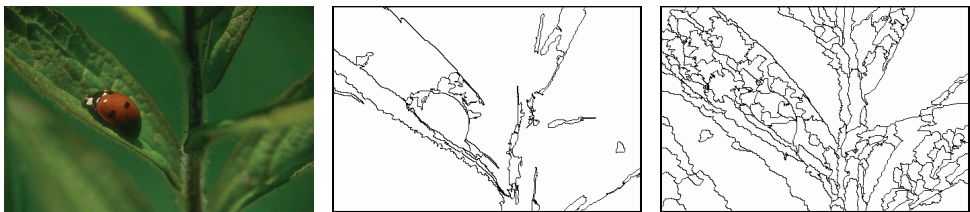


Fig. 2. An example of the segmentation of an image with two different algorithms. The first algorithm forms regions according to a multi-scale colour criterion while the second uses a local colour homogeneity criterion.

Once all the algorithms have been optimized, a third issue is to select the best one. However, when images of the application domain are highly variable, it remains quite impossible to achieve a good segmentation with only one tuned algorithm. Our objective is to make use of the extracted knowledge of context variations and parameter tuning to associate a segmentation action to each identified context.

Finally, in many computer vision systems at the detection layer, the goal is to separate the object(s) of interest from the image background. When objects of interest and/or image background are complex (e.g. composed of several subparts), a low-level algorithm cannot achieve a semantic segmentation, even if optimized. For this reason, a fourth issue is to refine the image segmentation to provide a semantically meaningful segmentation to higher vision modules.

Our final objective in this chapter is to show the potential of our approach through a segmentation task in a real-world application. The segmentation task we focus on is image segmentation in a biological application related to early pest detection and counting. This implies to robustly segment the objects of interest (mature white flies) from the complex background (rose leaves). Our goal is to demonstrate that the cognitive vision system coupled with our adaptive segmentation approach achieves a better detection rate of white flies than tuned with an *ad hoc* segmentation.

This chapter is structured as follows. Section 2 introduces the reader to image segmentation in the context of computer vision systems. We propose an overview on topics closely related to our problem. Section 3 details each step of our learning approach. Section 4 shows how the learnt segmentation knowledge is used to perform adaptive image segmentation. The next section is dedicated to the validation of the approach for a real world application: the segmentation step of a cognitive vision system dedicated to the recognition of biological organisms in static images. Concluding remarks and suggestions for future work are discussed in section 6.

2. Related work

In this section, we present some previous work related to image segmentation, segmentation performance evaluation, algorithm parameter optimization and algorithm selection.

2.1 Image segmentation

Several surveys of segmentation techniques have been published. Three of them (Pal & Pal, 1993; Skarbek & Koschan, 1994; Lucchese & Mitra, 2001) review about 300 publications giving a fair overview of the state-of-the-art in segmentation at the image-based processing level. Pal and Pal (Pal & Pal, 1993) mainly evaluate algorithms for grey-valued images and introduce three of the first attempts to exploit colour information. Skarbek and Koschan (Skarbek & Koschan, 1994) concentrate their survey on colour image segmentation. They classify the algorithms according to the underlying concepts of the homogeneity predicate and identify four categories: pixel-based, area-based, edge-based and physics-based approaches. Lucchese and Mitra (Lucchese & Mitra, 2001) also review exclusively colour segmentation approaches and use a similar categorization: feature space based, image domain based and physics based techniques. We can summarize these studies by making some important remarks, closely akin to the conclusions of (Skarbek & Koschan, 1994) in their survey:

1. General purpose algorithms are not robust and usually not algorithmically efficient.
2. All techniques are dependent on parameters, constants and thresholds which are usually fixed on the basis of few experiments. Tuning and adapting parameters is rarely performed.
3. As a rule, authors ignore comparing their novel ideas with existing ones.
4. As a rule, authors do not estimate the algorithmic complexity of their methods.
5. It seems that separating processes for region segmentation and for object recognition is the reason of failure of general purpose segmentation algorithms.
6. Several different colour spaces are employed for image segmentation. Nevertheless, no general advantage of one of the colour spaces with regard to the other colour spaces has been found yet.

2.2 Segmentation performance evaluation

Considering the increasing number of segmentation algorithms, the problem of performance segmentation evaluation becomes a primordial task. Two reasons motivate this statement: researchers must be able to compare their algorithm to other ones, and end-users must be able to choose an algorithm depending on the problem to solve. Usually, segmentation results are visually assessed by the algorithm's designer, which only allows subjective and qualitative conclusions on the algorithm performance. A generic method for the segmentation evaluation task does not exist, but many approaches have been proposed and can be classified into two principal classes: unsupervised methods and supervised methods (see Figure 3). The first class gathers the methods which do not require any *a priori* knowledge of segmentation results to evaluate. Their principle consists in estimating empirical criteria based on image statistics. The second class groups together evaluation methods based on *a priori* knowledge as a reference segmented image, usually named a ground truth (GT). A good survey of all these methods can be found in (Zhang, 1996) and in (Rosenberger et al., 2005).

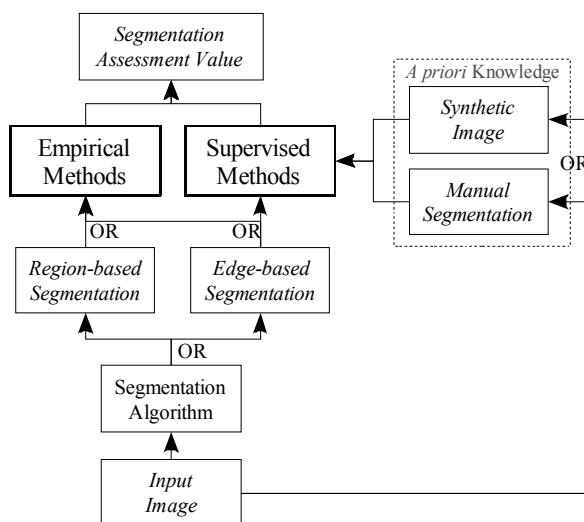


Fig. 3. Segmentation evaluation diagram starting from an input image and returning a segmentation assessment value

2.2.1 Unsupervised methods

The major advantage of unsupervised methods is that they do not require the intervention of an expert, just the definition of a metric of quality/discrepancy measure by the user is needed. Thus, these methods are totally automatic. However, defining a metric that could match all the segmentation objectives defined by the user is not a tricky task. Hence, quality measures are at best heuristic, since no specific knowledge of object(s) to segment is available. This tends to consider unsupervised performance evaluation method not very pertinent. Among the variety of proposed discrepancy measures, we can cite the well-known Rosenfield, Borsotti, Rosenberg or Charbrier criteria. A recent survey of these unsupervised methods can be found in (Zhang, 2008).

2.2.2 Supervised methods

Reference segmentations are achieved generally by hand or by generating synthetic images. In the last case, the ground truth data are objective and precise, in the contrary of subjective and imprecise hand-made expert drawing. These methods try to determine how far the actually segmented image is from the reference image in a quantitative manner, e.g. based on the number of misclassified pixels versus the reference segmentation. There are also a variety of discrepancy methods for the supervised evaluation of image segmentation. Some interesting ones can be found in (Yasnoff et al., 1977), (Everingham et al., 2002), and (Mezaris et al., 2003). The use of a ground truth is double-edged: it makes this class of methods potentially the most general and the less biased but this also supposes that ground truths are easily available. From this study, it also clearly appears that multi-objective methods yield better results than stand-alone methods (edge-based or region-based). However, the manner to combine measures remains an issue.

If we take a look at the number of publications around the segmentation evaluation problem, we can see that at present, this number is about one thousand concerning the segmentation algorithms, one hundred concerning the evaluation methods, and does not raise ten concerning the comparison of evaluation methods. If more efforts have been recently put on segmentation evaluation, it is still difficult to define wide-ranging performance metrics and statistics. Several explanations justify this limitation: (1) no common mathematical model or general strategy for evaluation is available especially for analytic methods; (2) no single evaluation can cover all aspects of segmentation algorithms; (3) appropriate ground truths are hard to determine objectively. Then, to overcome such limitations, potential research directions may explore methods combining multiple metrics in an effective manner (e.g., using learning) and methods considering the final goal of the segmentation.

Research is currently underway in terms of using these metrics as a mean to optimize parameters within a segmentation algorithm or to select the best adapted algorithm. This involves using an optimization procedure which is also a challenge in the context of image segmentation. The next section discusses this issue.

2.3 Algorithm parameter optimization

In this section, we relate some work dealing with segmentation algorithm parameter optimization. All the following approaches rely on three independent components: a segmentation algorithm with its free-parameters to tune, a segmentation quality assessment function and a global optimization algorithm as seen in Figure 4.

Researchers have experienced many segmentation optimization approaches during the last decade. Almost all of the free derivative optimization techniques have been tested. Interesting frameworks can be found in (Bahnu et al., 1995; Peng & Bahnu, 1998; Mao et al., 2000; Cinque et al., 2002; Gelasca et al., 2003; Pignalberi et al., 2003; Abdul-Karim et al., 2005). In the worst case, results of optimized segmentations are equivalent to the ones obtained with default parameters. In most of the cases, segmentation quality is improved and time spent to tune algorithms is drastically reduced. The authors present their frameworks as generic by nature and then widely applicable. This affirmation is well-founded in an analytical point of view since the three main components are considered separately. Nonetheless, each described framework has been set up for a particular segmentation task where the fitness function has been specifically elaborated for the

application using implicit domain knowledge. Thereby, it has not been proved how the fitness function can affect the performance of the optimization. Moreover, if authors have often assessed their optimization methods against default segmentations, they did not make any quantitative evaluation regarding to other optimization techniques. A comparative study of optimization algorithms has to be done.

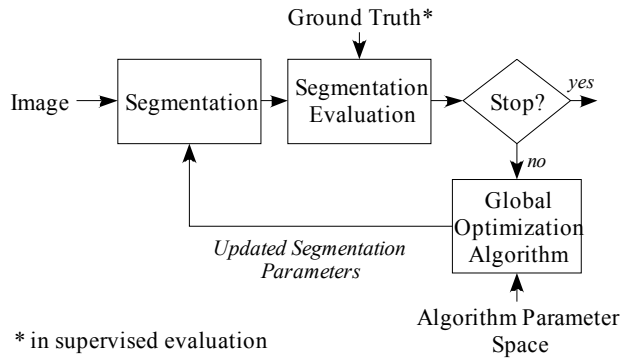


Fig. 4. The segmentation parameter optimization framework

2.4 Algorithm selection

In this section, we focus on the algorithm selection problem. Here, the goal is not to find the best parameter setting but rather to find the most suitable algorithm among several ones for a given segmentation task. Due to the still increasing number of algorithms, this problem has taken a big interest during the last decade. Basically, researchers tackle the problem with two different philosophies: model representation approach versus expert system approach.

In (Xia et al., 2005), the authors make the assumption that the choice of a segmentation algorithm can be predicted from a global feature vector. In other words, this means that a relationship between algorithm behaviours and global variations of image characteristic can be established (by means of learning techniques). The principal drawback is that the training process is imitated by the user assessment reliability. The task of visual algorithm ranking is time-consuming and then hardly conceivable in the case of large image and algorithm sets. As depicted by the authors, objective performance evaluation criteria (i.e. automatic) should be investigated to free users from the tedious training stage. In (Zhang & Luo, 2000), the authors propose a framework for automatic algorithm selection based on knowledge driven hypothesis-and-test optimization model. An expert system is designed to use evaluation, heuristic, and high-level knowledge (as *a priori* restrictions about domain dependent object features) to segment an image with the best adapted segmentation algorithm.

Globally, the two approaches rely on strong hypothesis concerning their field of applications: variations between images must be easy to model, algorithm behaviours within the images must be well-established, and high-level knowledge of objects to segment must be provided as a key-element of the performance evaluation. Actually, the lack of theory on segmentation rules out these approaches to be universally applicable. Indeed, application domains with image variations difficult to model disable the model representation approach, and the expensive knowledge acquisition task needed to build

expert systems limits their applicability. We can add that the model representation approach appears to be more realistic in a computing point of view as compared to expert systems.

2.5 Conclusion

We have reviewed the segmentation task in the field of computer vision systems. If researchers agree that segmentation is one of the fundamental problems in computer vision, the efforts devoted to cope with this issue since the last four decades have still not led to a unified solution. Most of the vision systems are application dependent and their segmentation step is based on heuristic rules for, as example, the tuning of algorithm parameters. It is, however, well-established that such *a priori* knowledge is determined by domain experts from the context in which the segmentation takes place. Hence, the generalization to other domain of application is strongly limited. Nonetheless, it appears that the recent cognitive vision approach (ECVISION, 2005) has identified some avenues of researches to cope with these limitations, as integration of machine learning techniques into the knowledge acquisition task.

3. Supervised learning for image segmentation

In this section, we present our cognitive vision approach to image segmentation. We have defined in section 1 the expectations of the segmentation task in computer vision systems (context adaptation, algorithm selection and tuning). We have seen in section 2 that these challenging issues have been tackled by many different approaches. Our goal is to propose a methodology that takes the best of each approach.

In the context of cognitive vision, we propose a framework with a reusability property to ease the set up of the segmentation task in vision systems. More precisely, our framework does not require image segmentation skills: the complexity of this tricky task is hidden by means of automatic algorithm parameter tuning and segmentation assessment. Moreover, the acquisition of the segmentation knowledge is made convenient by user-friendly interactivity. The second property of cognitive vision we are aiming at is the property of genericity. In our framework, the different components are not application dependent. Consequently, this framework can be used with different segmentation algorithms and for different real-world applications. The third cognitive property of our framework is its adaptation faculty to image content and to application needs. To this end, we use learning techniques for context adaptation, algorithm selection and parameter tuning.

3.1 Overview

Our framework consists of two stages: a learning stage and an adaptive segmentation stage. The framework relies on training data composed of manual segmentations of the training images with semantic region annotations. The learning stage extracts the segmentation knowledge from the training data by means of:

- a data mining module to extract and learn contextual variations,
- an optimization procedure for automatic segmentation parameter tuning,
- a learning module for context adaptation (i.e. to associate a segmentation action to each identified context),
- a learning module for semantic segmentation; the goal is to train region classifiers with respect to the annotated manual segmentations of the training images.

The learning stage is sketched in Figure 5. The module for adaptive image segmentation relies on the learnt segmentation knowledge. It will be described in section 4. The following sections details each step of the learning stage.

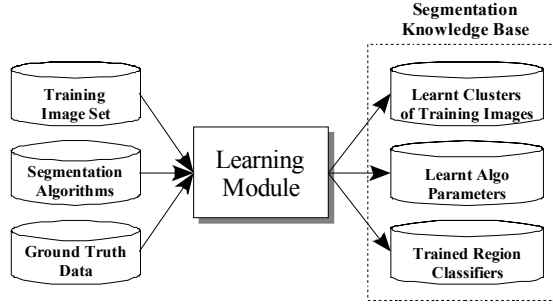


Fig. 5. The learning module for adaptive image segmentation.

3.2 Data mining for learning image contextual variations

Our strategy for algorithm selection is to tackle the problem *a priori* of the segmentation. In this case, the goal is not to directly select the algorithm depending on its relative performance evaluation but depending on the image to segment. Usually, variations between images lead to a variability in the segmentation. As a consequence, similar images should be segmented with the same algorithm and different images should be segmented with different algorithms or different parameter settings. These variations can be induced by changes in background appearance, changes in illumination source, or changes in imagery device configuration. The goal is to identify the different situations leading to different segmentation configurations. To this end, we define the context of an image as the quantitative representation of its local and global characteristics. Practically, the context is described by a d -dimensional feature vector $v(I)$ extracted from the whole image (e.g., a colour histograms). In our experiments, we have used a Density-Based Spatial clustering algorithm called DBScan proposed by Ester et al. (Ester et al., 1996) to identify the image clusters. This algorithm is well-adapted for clustering noisy data of arbitrary shape in high-dimensional space as histograms. Starting from one point, the algorithm searches for similar points in its neighbourhood based on a density criterion to manage noisy data. Non clustered points are considered as 'noise' points. The runtime of the algorithm is of the order $O(n \log n)$ with n the dimension of the input space. DBScan requires only one critical input parameter, the *Eps*-neighbourhood, and supports the user in determining an appropriate value for it. A low value will raises to many small clusters and may also classify a lot of points as noisy points, a high value prevents from noisy point detection but produces few clusters. A good value would be the density of the least dense cluster. But it is very hard to get this information on advance. Normally one does not know the distribution of the points in the space. If no cluster is found, all points are marked as noise. In our approach, we set this parameter so as to have at the most 15% of the training images classified as 'noise' data. We denote κ a cluster of training images belonging to the same context θ . The set of the n clusters is noted $K = \{\kappa_1, \dots, \kappa_n\}$ and the corresponding context set $\Theta = \{\theta_1, \dots, \theta_n\}$. Once the clustering is done, the internal data structures (here *R*-trees) and the DBScan parameters (*Eps*-neighbourhood, cluster IDs, etc.) are learnt.

3.3 Learning for segmentation parameter tuning

In this section, we detail our parameter optimization framework. The goal is to optimize the parameterisation of segmentation algorithms according to ground truth segmentations of the training images. For this task, the user must provide:

1. Manual segmentations of the training images with closed outlined regions.
2. Segmentation algorithms with their free parameters, i.e. the sensitive parameters to be tuned, as well as their range values. This kind of knowledge is often given by the algorithm's author.

3.3.1 Formalization of the optimization problem

Let I be an image of the training image set \mathfrak{I} , G_I be its ground truth (e.g. a manual segmentation), A be a segmentation algorithm and p^A vector of its free parameters. The segmentation of I with algorithm A is defined as $A(I, p^A)$. We define the segmentation quality E_I^A with the assessment function ρ as follows:

$$E_I^A = \rho(A(I, p^A), G_I) \quad (1)$$

The value E_I^A is an assessment value of the matching between the segmentation and the ground truth. This can be goodness or a discrepancy measure.

The purpose of our optimization procedure is to determine a set of parameter values \hat{p}_I^A which minimizes/maximizes:

$$\hat{p}_I^A = \arg \min_{p^A} / \max_{p^A} \rho(A(I, p^A), G_I) \quad (2)$$

The final assessment value \hat{E}_I^A and the optimal parameter set \hat{p}_I^A make a pair sample noted $(\hat{p}_I^A, \hat{E}_I^A)$. This pair forms the segmentation knowledge for the image I and the algorithm A . The set of all collected pairs constitutes the segmentation knowledge S set such that:

$$S = \bigcup_{I \in \mathfrak{I}} (\hat{p}_I^A, \hat{E}_I^A) \quad (3)$$

One key-point of this optimization procedure is the definition of the assessment function ρ . The quality of the final result varies according to this fitness function. The choice of a segmentation performance evaluation metric is hence fundamental. It is discussed in the next section.

3.3.2 Definition of the segmentation performance evaluation metric

As stated in section 2.2, it is not obvious to select a performance evaluation metric because no single metric can cover all aspects of segmentation algorithms. We propose to use a boundary-based metric and to evaluate the segmentation in terms of both localization accuracy and the shape accuracy of the extracted regions. The biggest advantage of boundary-based metrics against region-based metrics is their lower computational cost. It is always faster to count and compare some boundary pixels than a lot of region pixels. This metric is broadly usable since it mainly relies on generic concepts (false and missed boundary pixel rates).

The region boundary set for the ground truth and for the segmentation result are noted B_I^G and B_I^A respectively. Two types of errors are considered: missing boundary rate e_m^B and

false boundary rate e_f^B . The former, e_m^B , specifies the percentage of the points on B_I^G that are mistakenly classified as non-boundary points; while the latter, e_f^B , indicates the percentage of the points in B_I^A that are actually false alarms. Therefore,

$$e_m^B = \frac{|T_1|}{B_I^G} \text{ and } e_f^B = \frac{|T_2|}{B_I^A} \quad (4)$$

where

$$T_1 = \{x \mid (x \in B_I^G) \wedge (x \notin B_I^A)\} \\ \text{And } T_2 = \{x \mid (x \in B_I^A) \wedge (x \notin B_I^G)\} \quad (5)$$

and $|\cdot|$ is the cardinal operator.

We define the segmentation quality E_I^A with the assessment function as follows:

$$E_I^A = \rho(B_I^A, B_I^G) = \frac{1}{2}(e_m^B + e_f^B) \quad (6)$$

with $E_I^A \in [0,1]$.

The value $E_I^A = 0$ indicates perfect boundary pixel matching between the segmentation result and the ground truth when using algorithm A. The value $E_I^A = 1$ indicates that all pixels are misclassified. However, it is easy to show that this metric comes up against unsuited response to under-segmented results, as illustrated in Figure 6. Segmentation in panel (a) shows two regions with a quite good ground truth overlap, only three pixels are misclassified. In the panel (b), the segmentation shows only one region and the quality score is logically less than in (a). In the last panel (c), two regions are present but the centre region badly overlaps the corresponding ground truth centre region. In opposition with visual assessment, the segmentation quality is worst than in Figure 6(c).

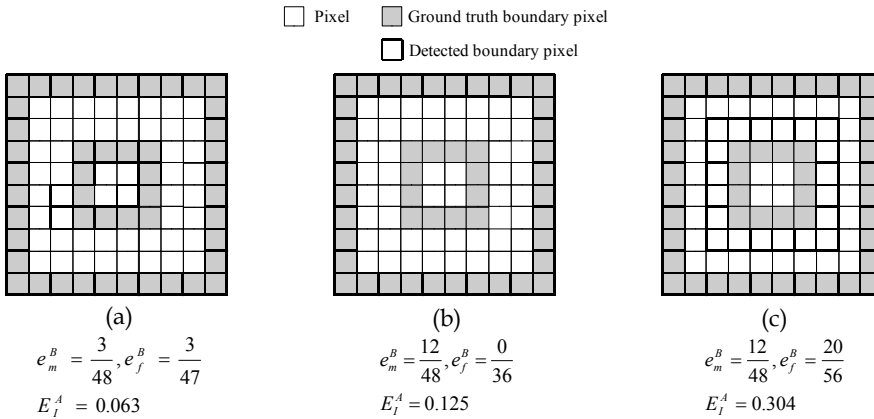


Fig. 6. Limitation of the segmentation evaluation metric when weighting terms (w_m^B and w_f^B) are not used.

The metric is improved by introducing two weighting terms w_m^B and w_f^B which quantify the average distance between misclassified points to the ground truth boundary such that:

$$w_m^B = \frac{1}{|T_1|} \sum_{x \in T_1} \text{dist}(x, \hat{x}_I^A) \quad (7)$$

with \hat{x}_I^A the closest pixel to x belonging to B_I^A , and

$$w_f^B = \frac{1}{|T_2|} \sum_{x \in T_2} \text{dist}(x, \hat{x}_I^G) \quad (8)$$

with \hat{x}_I^G the closest pixel to x belonging to B_I^G ; $\text{dist}(x_1, x_2)$ is the Euclidean distance between two pixels $x_1(u, v)$ and $x_2(u, v)$ in a 4-neighbourhood such that:

$$\text{dist}(x_1, x_2) = \sqrt{(x_1(u) - x_2(u))^2 + (x_1(v) - x_2(v))^2} \quad (9)$$

Since w_m^B and w_f^B have no fixed upper bounds, the normalization factor is useless and the segmentation quality measure becomes:

$$\begin{aligned} E_I^A &= w_m^B \times e_m^B + w_f^B \times e_f^B \\ &= \frac{1}{B_I^G} \sum_{x \in T_1} \text{dist}(x, \hat{x}_I^A) + \frac{1}{B_I^A} \sum_{x \in T_2} \text{dist}(x, \hat{x}_I^G) \end{aligned} \quad (10)$$

The search of \hat{x}_I^A (resp. \hat{x}_I^G) is made easier by the use of a distance map (Maurer & Raghavan, 2003) computed from B_I^A (resp. B_I^G). This operation is exemplified in Figure 7.

By taking back the example in Figure 6 with the new definition of the evaluation metric, the values of E_I^A for the cases (a), (b), and (c) are respectively 0.168, 0.75, and 0.679, yielding a good correlation with a visual assessment.

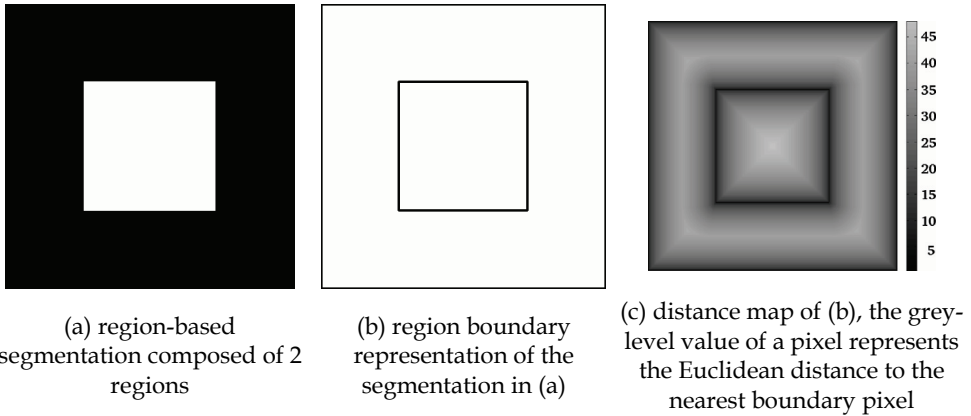


Fig. 7. An example of a distance map from a binary contour segmentation.

Once our performance evaluation metric is defined, the goal is now to minimize the segmentation error E_I^A in order to learn optimal segmentation parameters. This is the role of our closed-loop global optimization procedure.

3.3.3 Choice of the optimization algorithm

Of primary importance in this optimization procedure is finding an optimal segmentation parameter setting \hat{p}_I^A for each $I \in \mathfrak{I}$. We also aim at providing a good evaluation study of the tested optimization techniques in terms of performance versus computational cost and parameter setting. In the family of free derivative techniques, we propose the following criteria to assess the optimization algorithms:

1. Since the segmentation of an image is the most expensive process in the optimization loop, the number of maximum segmentation algorithm calls might be set as a parameter. Indeed, even if the ultimate goal of an optimization procedure is to find a global optimum, the computational cost should remain realistic.
2. The optimization algorithm must be able to converge whatever the evaluation profile, i.e. robust enough to find (quasi-)global optimum of various non-smooth functions.
3. The final quality of the optimization procedure should no be too dependent of the tuning of the optimization algorithm parameters, whatever the segmentation algorithm.

We have seen in our survey (see section 2.3) that several optimization techniques have been applied to tackle the segmentation optimization problem. Although all of them are suitable with our problem, no comparative study exists to help us in our choice. Thus, we have decided to focus on two techniques which are worth being compared. The first one is the Simplex algorithm (Nelder & Mead, 1965) and the second is a standard genetic algorithm (Goldberg, 1989) using non-overlapping populations and optional elitism. In one hand, simplex is easy to use, fast to converge, but requires to define a initializing strategy (starting point(s) and starting step) and do not guarantee to find a global optimum. In an other hand, genetic algorithms are robust but are slower to converge and their parameters must be set carefully.

After all pair samples $(\hat{p}_I^A, \hat{E}_I^A)$ have been extracted for all segmentation algorithms to test, the next step is to select and tune the one(s) which will be learnt for each identified context. The following section discusses our learning strategy for context adaptation.

3.4 Learning for context adaptation

The previous parameter optimization step allows us to objectively compare the segmentation algorithms with regards to their best performance scores. A straightforward strategy for the selection of an algorithm is thus to take the first best. Nevertheless, the problem becomes more difficult when the training images are heterogeneous, due for instance to global or local variations in the background. In this case, one segmentation algorithm could be the best adapted for the segmentation of a training image subset and another one for another subset. We propose to tackle this problem by associating one algorithm per subset. More precisely, we propose to rank the segmentation algorithms for each previously identified context. The context adaptation strategy can be formalized as follows:

$$\begin{aligned} f : R^d &\rightarrow S \\ v(I) &\mapsto (A, \hat{p}^A) \end{aligned} \tag{11}$$

However, it is impossible to continuously predict the algorithm behaviour according to image variations and therefore the function cannot be seen as a regression model. Our approach is to tackle this modelling problem by applying an unsupervised clustering of the training images to identify the different contexts, i.e. clusters of images having similar feature vectors. Then, for each cluster (i.e. images of the same context), segmentation algorithms are ranked and the best one is learnt. The best algorithm is the one performing the best average performance on the cluster. For each algorithm, a mean parameter set is computed as follows:

$$\bar{p}^A = \frac{1}{|\mathfrak{S}_A|} \sum_{I \in \mathfrak{S}_A} \hat{p}_I^A \quad (12)$$

where \mathfrak{S}_A is the subset of training images for which the algorithm A has obtained the best evaluation results among the other algorithms. Finally, for each training image of the cluster and each algorithm A tuned with \bar{p}^A , the segmentation quality is computed again. The algorithm having the best average performance on the training image set is finally selected. We obtain a discrete function F taking a context identifier θ as input and returning an algorithm A with a mean parameter setting \bar{p}^A such as:

$$\begin{aligned} F : \Theta &\rightarrow S \\ \theta &\mapsto (A, \bar{p}^A) \end{aligned} \quad (13)$$

This selection strategy comes to select the robustest algorithm based on objective comparisons, i.e. the algorithm which can deliver the best results for the cluster with a globally relevant parameter set. However, this straightforward ranking approach has two major drawbacks. First, by selecting only one algorithm and averaging its parameters, it reduces the previously extracted segmentation knowledge amount to one mean case. Second, even if the selected algorithm over performs the others in most of the cases, the parameter averaging can have disastrous effects on the algorithm performance.

The principal purpose of this strategy is to overcome the drawbacks of a pure global ranking strategy by dividing the solution space and by restricting the ranking process onto each subspace. The main advantage on ranking algorithms inside a subspace is that evaluation profiles are likely more correlated.

In this section, we have shown that the algorithm selection problem cannot be separated from the parameter tuning problem. This statement means that a solution to the algorithm selection issue is composed of both an algorithm and a parameter setting. We have described our twofold strategy for learning the algorithm selection based on image-content analysis and algorithm ranking. Starting from a training image set and segmentation algorithms, our approach first identifies different situations based on image-content analysis, then select the best algorithm with a mean parameter set for each identified context based on optimized parameter values. At the end of the learning process, contexts are learnt with their associated pairs (A, \bar{p}^A) .

3.5 Learning for semantic image segmentation

In this section, we propose an approach for semantic image segmentation based on high-level knowledge acquisition and learning. Even if the segmentation is optimized, low-level

segmentation algorithms cannot reach a semantic partitioning of the image. Thus, compared to the ground truth, some regions remain over-segmented, as illustrated in Figure 8. If we can assign the right label to each region, neighbouring regions with similar labels are merged and, as a consequence, the residual over-segmentation becomes invisible. This means to be able to map region features onto a symbolic concept, i.e. a class label. We use the example-based modelling approach as an implicit representation of the low-level knowledge. This approach has been applied successfully in many applications such as detection and segmentation of objects from specific classes e.g., (Schnitman et al., 2006; Borenstein & Malik, 2006). Starting from representative patch-based samples of objects (e.g., fragments), modelling techniques (e.g., mixture of Gaussian, neural networks, naive Bayes classifiers) are implemented to obtain codebooks or class-specific detectors for the segmentation of images. Our strategy follows this implicit knowledge representation and associates it with machine learning techniques to train region classifiers. The following subsections describe this stage in details.

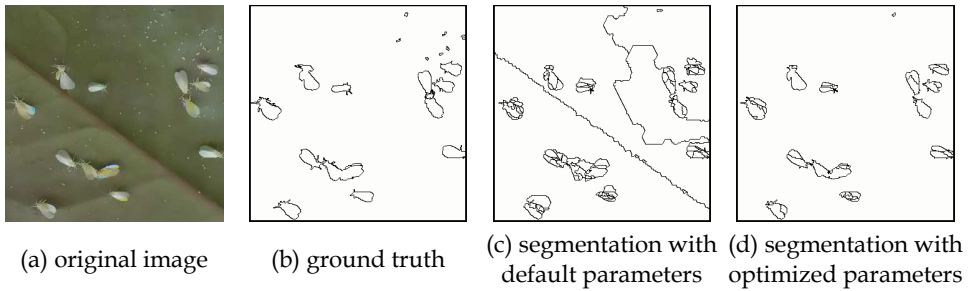


Fig. 8. An example of a parameter optimization loop. The final result (d) is not perfect since some regions are over-segmented with respect to the ground truth (b).

3.5.1 Class knowledge acquisition by region annotations

In our case, region annotations represent the high-level information. This approach assumes that the user is able to gather, in a first step, a representative set of manually segmented training images, i.e. a set that illustrates the variability of object characteristics which may be found. Then, the user must define a domain class dictionary composed of k classes as $Y = \{y_1, \dots, y_k\}$. This dictionary must be designed according to the problem objectives. For instance, y_1 = background class, y_2 = object class #1, and so on. Once Y is defined, the user is invited, in a supervised stage, to label the regions of the manually segmented images with respect to Y . From a practical point of view, an annotation is done with the help of a graphical user interface we have developed. This tool allows interacting with a region-based segmentation of an image by clicking into a region and by selecting the desired class label y (see Figure 9).

At the end of the annotation task, we obtain a list of labelled ground truth regions which belong to classes defined by the user. Since the segmentation result is not exactly the same than the manual segmentation, the next step is to map, for each training image, the labels of ground truth regions onto the regions of the region map R_I^A resulting from the segmentation of the image I with the selected algorithm A tuned with the parameter set \bar{p}^A , as described in section 3.4. The mapping is done by majority overlap such as for each region $r \in R_I^A$,

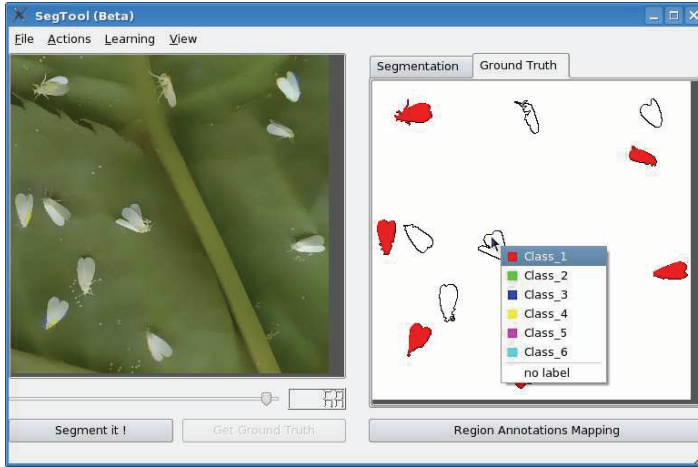


Fig. 9. Region annotations with the developed graphical tool.

$$y(r) = \begin{cases} y_i & | i = \arg \max H_r, \text{ if } \frac{\max h(r)}{|r|} > T \\ y_0, & \text{else} \end{cases} \quad (14)$$

with $|r|$ the number of pixels of the region r , T a threshold, and $H(r) = \{h_1(r), \dots, h_i(r), \dots, h_k(r)\}$ the label histogram of the region r such that for a pixel u and a label y_i , $h_i(r) = \text{card}\{u \in r \mid y(u) = y_i\}$, $i \in 1, \dots, k$.

If the ratio of the most represented class in the region does not reach the threshold T (here fixed at 0.8), the region label is set to $y_0 \notin Y$. This prevents from labelling badly segmented region as sketched in Figure 10.

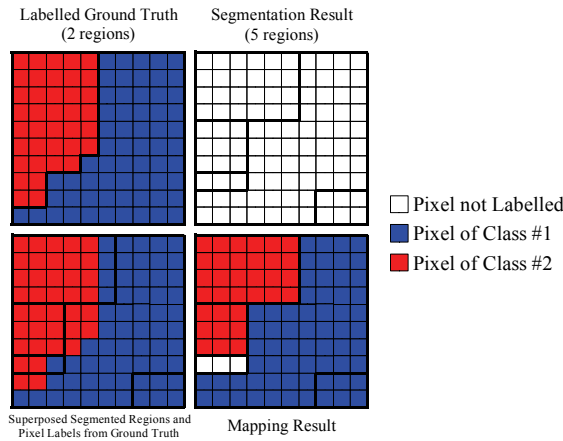


Fig. 10. Example of the mapping between a labelled ground truth regions and segmented regions.

We also denote the set of all region annotations

$$RA_{\mathfrak{I}} = \bigcup_{I \in \mathfrak{I}} \bigcup_{r \in R_I^A} \{y(r) \mid y(r) \neq y_0\} \quad (15)$$

and the set of all annotated regions $R_{\mathfrak{I}}$ such as:

$$R_{\mathfrak{I}} = \bigcup_{I \in \mathfrak{I}} \bigcup_{r \in R_I^A} \{r \mid y(r) \neq y_0\} \quad (16)$$

for each region, a feature vector $x(r)$ is extracted and makes with the label a pair sample noted $(x(r), y(r))$. The set of all collected pair samples from \mathfrak{I} constitute the training data set such as:

$$T_{\mathfrak{I}} = \bigcup_{I \in \mathfrak{I}} \bigcup_{r \in R_I^A} \{(x(r), y(r)) \mid y(r) \neq y_0\} \quad (17)$$

$T_{\mathfrak{I}}$ represents the knowledge of the semantic segmentation task and is composed, at this time, of raw information. In the following section, we address the problem of knowledge modelling by statistical analysis.

3.5.2 Segmentation knowledge modelling

The first step towards learning statistical models from an image partition is to extract a feature vector from each region. But which low-level features are the most representative for a specific region labelling problem? In more general terms, which features are useful to build a good model predictor? This fundamental question, referring to the feature selection problem, is a key issue for most of the class-based segmentation approaches.

Feature Extraction

When defining a set of features for classification problems, two approaches can be considered: a first approach aims at building relevant feature sets, while a second approach more focus on the usefulness of each feature. In the first case, the choice of relevant features mostly relies on knowledge of the domain. In the second case, the goal is clearly to select features useful for building a good predictor, even if some relevant features may be excluded. We propose a trade-off approach: starting from heuristically selected features we aim at training robust region classifiers. To this end, we combine generic features, such as colour and texture and apply a feature selection algorithm.

In our approach, colour histograms represent the colour information of each segmented region. Two parameters must be set: the colour space (cs) as RGB, HSV, or XYZ, on which the histogramming is applied, and the quantization parameter q which defines the number of bins. In our approach, we do not state *a priori* the relevance of one colour space against others as well as the best quantization level. We rather consider these variables as parameters of the feature selection problem.

Texture feature extraction techniques have received considerable attention during the past decades and numerous approaches and comparative studies have been presented (Reed & du Buf, 1993). The most commonly used are the grey-level co-occurrence matrices introduced by Haralick (Haralick, 1979), the Law's texture energy (Laws, 1980), and the Gabor multi-channel filtering (Jain & Farrokhnia, 1991). For the characterization of texture, we use oriented Gaussian derivatives (OGD) to generate rotation invariant feature vectors. OGD are

equivalent to the Gabor features but are computationally simpler. The basic idea is to compute the *energy* of a region as a steerable function. This energy is computed for different *power* channel, which are the result of convolving the region pixels with OGD filters of a specific order. As colour histograms, texture feature vectors depend on the parameter q .

The final feature vector representing a region is a concatenation of the feature vectors extracted from each cue. The feature extraction process is applied on each region of the annotated regions set R_3 so as to build the training data set T_3 .

Following our cognitive approach of the segmentation problem, we need to avoid manually selected and tuned algorithms. At the feature selection level, this means to be able to automatically select and tune the feature extraction algorithm.

Feature Selection

The feature selection is used to reduce the number of features, remove irrelevant, redundant, or noisy data, and it brings the immediate effects of speeding up and improving the prediction performance of learning models. Since feature selection is a fertile field of research, we refer the reader to surveys (Guyon & Elisseeff, 2003; Kohavi & John, 1997; Blum & Langley, 1997) as good starting literatures. The optimality of a feature subset is measured by an evaluation criterion. Feature selection algorithms designed with different evaluation criteria broadly fall into two categories: the *filters* and the *wrappers*. Filters select subsets of features as a pre-processing step, independently of the chosen predictor. Well-known methods dedicated to this purpose are basic linear transforms of the input features like Principal Component Analysis (PCA) and Fisher Linear Discriminant Analysis (LDA). Techniques based on iterative search are also widespread as sequential forward/backward algorithms (e.g. SFFS, SBS, ReliefF). Wrappers utilize the learning machine of interest (e.g., SVM, neural networks) as a black box to score subsets of features according to their predictive power. Consequently, wrappers are remarkably universal and simple. An interesting comparative study of such feature selection algorithms can be found in (Molina et al., 2002).

The feature selection approach we propose is derived from wrappers. Our goal is to find the best feature extractor configuration which minimizes the joint classification errors of the class predictors applied on the training data set T_3 . Unlike classical approaches, we act on the feature extractor parameters to generate different feature vectors, instead of reducing the feature vector itself. This approach is sketched in Figure 11. The two free parameters of our selected feature extractors are the colour space encoder for colour feature extractor, and the quantization level for both colour and texture feature extractors. The goal is to find the best combination able to induce the minimum region classification errors. The quality estimation is conducted via a cross-validation procedure which gives, for each region classifier, the classification Mean Square Error (MSE). A global MSE is then computed by averaging the MSE of each region classifiers.

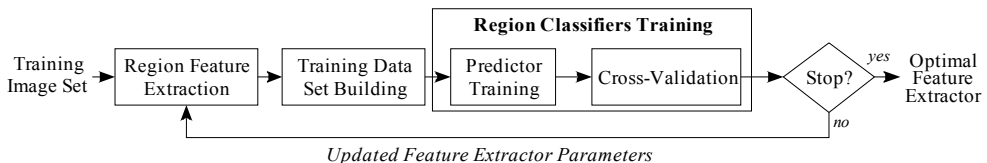


Fig. 11. Feature selection schema based on tuning of the feature extractor parameters.

We use an iterative search strategy to cover the value spaces of the two parameters q and cs . This technique guarantees to find a global optimal solution but is computationally expensive: first, it requires to run $M \times N \times O$ region classifier training procedures, with M the number of quantization levels (typically equals to 256), N the number of color spaces, and O the number of classifiers to train; second, when the value of q increases, so does the size of the feature vector. So, to avoid an unreasonable computational time, the choice of the training algorithm must take into account this computational constraint.

Training Algorithm for Class Modelling

After extracting a feature vector for each region of the training data set, the next step is to model the knowledge in order to produce region classifiers (one classifier per class). For a feature vector $x(r)$ and a class y_i ,

$$c_i(r) = p(y(r) = y_i | x(r)) \quad (18)$$

with $c_i(r) \in [0,1]$ is the estimated probability associated with the hypothesis: "feature vector $x(r)$ extracted from region r is a representative sample of the class y_i ". The set of these trained region classifiers is noted $C = \{c_1, \dots, c_k\}$.

A variety of techniques have been successfully employed to tackle the problem of knowledge modelling such as naives Bayes networks, decision trees or support vector machine (SVM). We propose to use SVM (Burges, 1998) as a template-based approach. SVM are known to be efficient discriminative strategies for large scale classification problems such as in image categorization (Chen & Wang, 2004) or object categorization (Huang & LeCun, 2006). SVM yields also state-of-the-art performance at very low computational cost. SVM training consists of finding an hyper-surface in the space of possible inputs (i.e. feature vectors labelled by +1 or -1). This hyper-surface will attempt to split the positive samples from the negative samples. This split will be chosen to have the largest distance from the hyper-surface to the nearest of the positive and negative samples.

We adopt a one-vs-rest multi-class scheme with probability information (Wu et al., 2004) to train one region evaluator per class. We use SVM with radial basis function as region classifiers. There are two parameters while using RBF kernels: C (penalty parameter of the error term) and γ (kernel parameter). It is not known beforehand which C and γ are the best for one problem; consequently some kind of model selection (parameter search) must be done. To fit the C and γ parameters, we adopt a grid-search method using 5-fold cross-validation on training data. Basically, pairs of (C, γ) are tried and the one with the best cross-validation accuracy is picked. This straightforward model selection efficiently prevents over-fitting problems. The model selection is wrapped in the feature selection schema with which it shares the cross-validation step. The training stage ends up when all combinations of $((q, cs), (C, \gamma))$ have been tested. The one giving the lowest global classification error is picked and the region classifiers are trained a last time with this configuration.

3.6 Conclusion

In this section, we have presented our learning approach for adaptive image segmentation. We have detailed each step of the learning module for context adaptation, algorithm parameter tuning, and semantic image segmentation. The algorithm parameterisation issue is tackled with a generic optimization procedure based on three independent components. We have designed our performance evaluation metric to be broadly applicable and with a low computational cost. It allows assessing a large variety of segmentation algorithms and

only relies on manual segmentations. However, further experiments need to be done to assess the performance and the accuracy of the two optimization algorithms (the Simplex algorithm and a Genetic Algorithm). The final step of the learning module is to train region classifiers to refine the segmentation according to semantic region labelling. In this task, the user must annotate the regions of the manually segmented images with class labels. Our approach is based on the discriminative power of the SVM Classifiers to ground low-level region features into symbolic classes. We have also proposed an unsupervised method for the learning of SVM and region feature extractor parameters. The goal is to optimize the performance of the classifiers without the help of the user.

4. Adaptive image segmentation

The originality of our approach is to combine bottom-up segmentation and a top-down process of region labelling in a complementary manner: in a first step, segmentation is optimized by dynamic algorithm selection and parameter tuning. Then, the bottom-up segmentation is refined thanks to region labelling to achieve the expected semantic segmentation. A new image (i.e. not belonging to the training set) segmentation is achieved by the adaptive image segmentation module in four steps (see Figure 12) using the segmentation knowledge base (learnt clusters of training images, learnt parameters, and trained region classifiers):

1. Context Prediction: a global feature vector is extracted from the image. The feature vector is classified among the previously identified clusters. The classification is obtained by assessing the distance of the feature vector to the cluster centres.
2. Algorithm selection: from the identified context, the corresponding segmentation algorithm with learnt parameters is selected.
3. Bottom-up segmentation: the image is segmented using the selected algorithm. This algorithm is tuned with the learnt parameters specific to the identified context.
4. Semantic segmentation: for each region of the segmented image, features are extracted and given as input to the region classifiers. The most probable label is assigned to the region. The final labelled partition representing the semantic segmentation of the image is returned to the user.

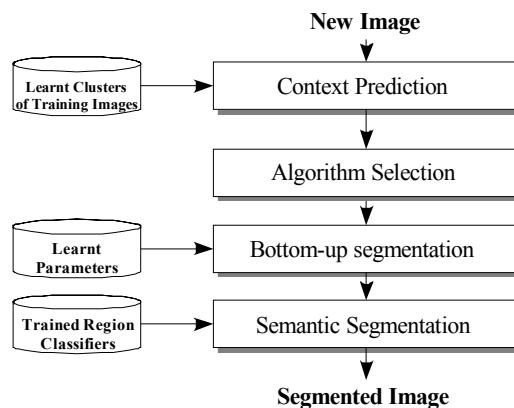


Fig. 12. Adaptive segmentation of an input image based on algorithm selection, parameter tuning, and region labelling.

For a new incoming image I not belonging to the training set, a feature vector is first extracted then classified into a cluster. The classification is based on the minimization of the distance between the feature vector and the cluster set $\{\kappa_i\}$ as follows:

$$I \in \theta_i \Leftrightarrow v(I) \in \kappa_i \mid i = \arg \min_{i \in [1,n]} \text{dist}(v(I), \kappa_i) \quad (19)$$

The pair (A, \bar{p}^A) associated with the detected context θ_i is returned.

Once the algorithm is selected and tuned, the image is segmented. For each region, a feature vector is extracted using the optimized (\hat{q}, cs) parameter set and given as input to each trained region classifiers c_i . Classes are scored according to the classifier responses $\{c_i(r)\}$ and finally, the returned label $y(r)$ is such as:

$$y(r) = \arg \max_{ii} c_i(r) \quad (20)$$

When all regions are labelled, neighbouring regions with the same label are merged to form a semantic partitioning of the image. This final segmentation is returned to the user.

5. Experiment for adaptive image segmentation

We present below the experimentations we have conducted to assess our framework. The first application we focus on is the segmentation of biological objects in their natural environment. More precisely, the goal is to segment white flies on rose leaves (see Figure 13). The images are acquired from a flatbed scanner. White flies are very small objects (2mm), their wings are semi-translucents, and they can be seen from different points of view. Rose leaves are highly textured with many veins and present various appearances.

5.1 Segmentation algorithms

In this section, we briefly describe the segmentation algorithms we used for our experiment. Our set is composed of algorithms reflecting different segmentation strategies as developed in section 2.1 namely region growing, split-and-merge, watershed, or thresholding techniques. The Table 1 summarizes these algorithms and gives important information concerning their free parameter with their ranges and default values provided by the algorithms' authors.

5.2 Parameter optimization assessment

Before assessing the optimization procedure, we illustrate the optimization problem with some examples of evaluation profiles. We present 1D and 2D profiles for the different segmentation algorithms (except the CWAGM which has a parameter space in R^3) for the four training images of the Figure 13. The best segmentation quality correspond to an assessment value $E_I^A = 0$. Concerning the CSC algorithm (see Figure 14), the shapes of the curves are similar for the four images and present a global minimum which falls in the same part of the parameter space. The global optima for the SRM algorithm (see Figure 15) are found in a very narrow band of the parameter space. Many local optima characterize the curves of the EGBIS algorithm (see Figure 16). The thresholding algorithm behaviour is more straightforward regarding to the obtained curves (see Figure 17). Globally, two

performance levels are revealed where good performances are achieved for a large range of the parameter values. However, the global optimum is more difficult to see since the difference between the good performance level (in blue) and its level is very thin. From these observations, we can conclude that the evaluation profiles are not always convex hulls and their granularity can depend on the image. Since the Simplex algorithm does not guarantee to obtain a global optimum, we divide each parameter space into three sub-spaces and run an optimization on each sub-space. This means that 3^N optimization loops are run for a segmentation algorithm with N free-parameters. Table 2 present the optimization results of the five segmentation algorithms in terms of segmentation performance. Globally, all the algorithms reach a good level except the EGBIS algorithm, as shown in Figure 18. This result is due to the fact that this algorithm is sensitive to small gradient variations. As expected, the EGBIS has a big standard deviation (due to the presence of many local optima) whereas the thresholding one is low (due to its straightforward behaviour). We have also compared the performances of the optimization algorithms (the Simplex and the GA) with a systematic search method (third part of Table 2). By systematic, we mean an iterative search throughout the whole parameter space with a fixed sampling rate. The sampling rate depends on the dimensionality of the parameter space. The global performances of the three methods are similar with a very little advantage to the Simplex.

Algorithm	Free Parameter	Range	Default Value
CSC (Prieze et al., 2002) Color Structure Code	t : region merging threshold	5.0-255.0	20.0
SRM (Nock & Nielsen, 2004) Statistical Region Merging	Q : coarse-to-fine scale control	1.0-255.0	32.0
EGBIS (Felzenszwalb & Huttenlocher, 2004) Efficient Graph-Based Image Segmentation	σ : smooth control on input image k : colour space threshold	0.0-1.0 0.0-2000.0	0.50 500.0
Hysteresis thresholding	T_{low} : low threshold	0.0-1.0	-
	T_{high} : high threshold	0.0-1.0	-
CWAGM (Alvarado Moya, 2004) Color Watershed-Adjacency Graph Merge	m : region merging threshold	0.0-200.0	100.0
	n : min. region number	1.0-100.0	10.0
	p : min. probability for watershed threshold	0.0-1.0	0.45

Table 1. Components of the segmentation algorithm bank, their names, and parameters to tune with range and author's default values.

To decide between the three different methods, we have compared them by considering their computational cost as described in Table 3. The systematic search is obviously the most costly method. The Simplex is the fastest method to converge apart from the CWAGM algorithm. According to the previous performance score tables, the simplex is definitively the best algorithm to optimize low dimensional parameter spaces in a few numbers of iterations. For segmentation algorithms with more than two free-parameters, the Genetic Algorithm should be preferred, requiring less iteration for the same level of performance. Note that we have limited the number of iterations – mainly for computational cost

reasons—for the systematic search method to 2550 for the EGBIS algorithm and to 1250 for the CWAGM algorithm, respectively. These two algorithms are relatively slow compared to the others and the parameter space to explore is really huge, particularly for the CWAGM.

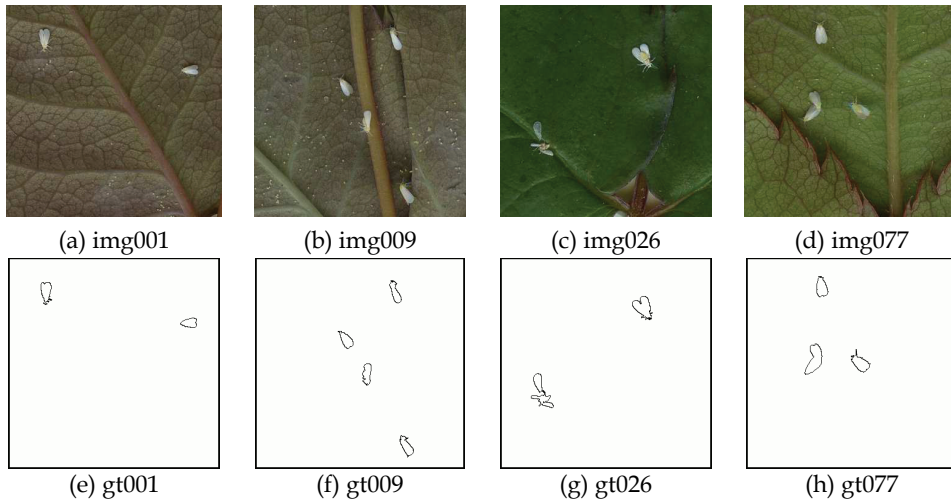


Fig. 13. Four representative training images and associated ground truth segmentations used in figure 14 to figure 17.

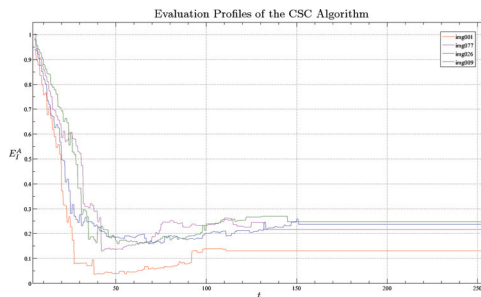


Fig. 14. Evaluation profiles of the CSC algorithm applied on the four training images presented in Figure 13.

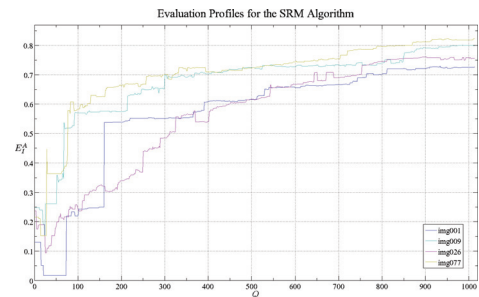


Fig. 15. Evaluation profiles of the SRM algorithm applied on the four training images presented in Figure 13.

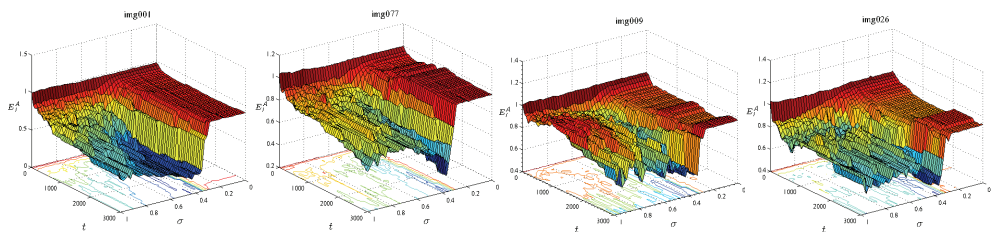


Fig. 16. Different evaluation profiles of the EGBIS algorithm applied on the four training images presented in Figure 13. t and σ are the two free parameters.

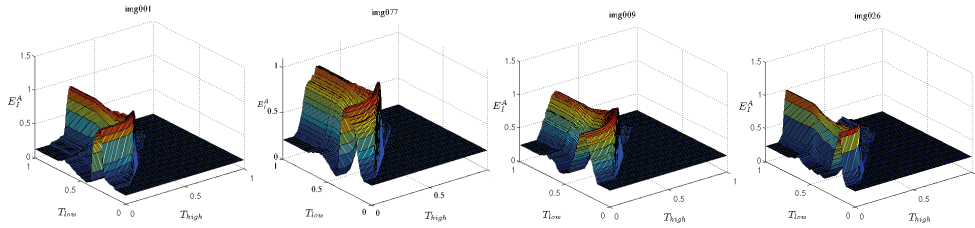


Fig. 17. Different evaluation profiles of the Hysteresis thresholding algorithm applied on the four training images presented in Figure 13. T_{low} and T_{high} are the two free parameters.

Algorithm	$E_f^A = 0$ using Simplex / GA / Iterative search			
	min	max	mean	std
CSC	0.00 / 0.00 / 0.00	0.50 / 0.46 / 0.46	0.14 / 0.13 / 0.13	0.11 / 0.10 / 0.10
SRM	0.00 / 0.00 / 0.00	0.52 / 0.48 / 0.48	0.13 / 0.12 / 0.12	0.11 / 0.10 / 0.10
THRESH	0.00 / 0.00 / 0.00	0.35 / 0.35 / 0.35	0.11 / 0.11 / 0.11	0.09 / 0.09 / 0.09
EGBIS	0.06 / 0.12 / 0.13	0.73 / 0.71 / 0.71	0.37 / 0.37 / 0.39	0.14 / 0.14 / 0.14
CWAGM	0.00 / 0.00 / 0.00	0.44 / 0.44 / 0.46	0.12 / 0.12 / 0.19	0.09 / 0.09 / 0.08

Table 2. Statistics on the optimization performances for the training image set using the Simplex algorithm, the genetic algorithm and the systematic search.

Algorithm	Mean number of iterations		
	Systematic search	Genetic algorithm	Simplex algorithm
CSC	1000	733	83
SRM	1000	734	82
THRESH	10000	840	404
EGBIS	2550	840	497
CWAGM	1250	840	1821

Table 3. Computational cost of each optimization method.

The number of iterations is also dependent of the parameterisation of the optimization algorithm. For the Simplex algorithm, it mainly depends on the *maxCalls* parameter which specifies the maximum allowed number of calls of the fitness function in an optimization loop. Figure 18 (left) shows the influence of this parameter on the convergence accuracy. We start the test on the img001 with *maxCalls* set to 3 (minimum allowed by the algorithm) and increase it up to 80. For a one-dimensional parameter space, this means that the total number of iterations will be between 9 (3×3) and 240 (3×80), for a two dimensional space between 27 ($3^2 \times 3$) and 720 ($3^2 \times 80$), and so on. The study of the graph brings us to several conclusions. The dimensionality of the parameter space to explore has to be taken into account for the setting of *maxCalls* but excessive values are useless. The study also reveals that the parameter space is not explored in the same way, depending on the segmentation algorithm. Indeed, some algorithms have parameter subspaces which induce flat evaluation profiles, as for instance the thresholding algorithm. In these sub-spaces, the Simplex

converges in a few numbers of iterations. The same study is done for the GA and the results are graphically reported in Figure 18 (right). We decide to assess the GA sensitivity to the initial population size. The number of initial points is here independent of the segmentation algorithm and varies between 20 and 840. The same conclusions can be drawn. We just can add that the EGBIS algorithm brings some problem to the GA which falls in many local optima (peaks of the EGBIS curve in Figure 18 (right)).

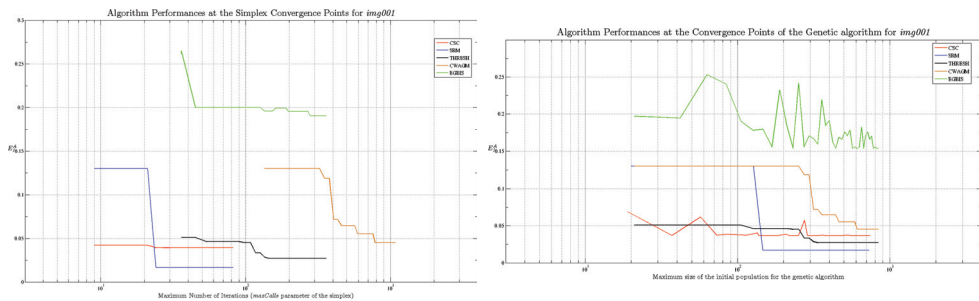


Fig. 18. Convergence accuracy of the Simplex algorithm by varying the *maxCalls* parameter and convergence accuracy of the GA by varying the initial population size.

5.3 Algorithm selection

We applied the DBScan (Ester et al., 1996) algorithm to cluster the 20 training images as described in section 3.2. We obtain two clusters of 10 images (see Figure 19 for two examples). Visually, the first cluster corresponds to the back side images of the scanned rose leaves and the second cluster to the front side images. For each cluster, mean parameter sets of the five segmentation algorithms are computed w.r.t. their performance scores. The segmentation performances of the tuned algorithms are evaluated on each training image sub set. The tuned algorithm which gets the best mean performance score for each cluster is elected. Before the last ranking step, the best algorithm for the first cluster was the Hysteresis thresholding algorithm and the best for the second cluster was the CSC algorithm. After the last ranking step, the CSC algorithm was found as the best one for the two clusters but with different parameter sets. This means that even if the thresholding algorithm performs better in individual cases, the CSC algorithm is more robust than the thresholding algorithm when tuned with a mean parameter set.

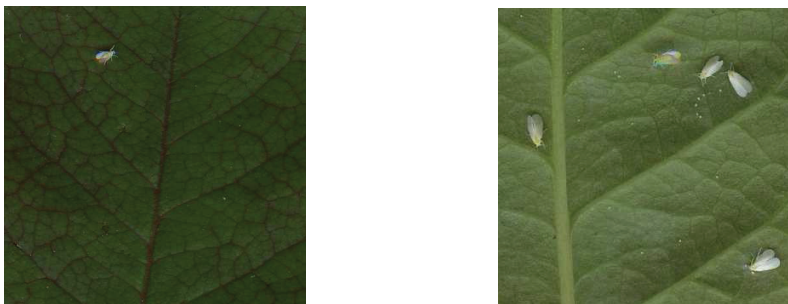


Fig. 19. Examples of images for the two identified clusters. Left = cluster 1 (front side of the leaves), right = cluster 2 (back side of the leaves).

5.4 Semantic segmentation performance assessment

For each identified image cluster, region labels of annotated manual segmentations are mapped into regions of the segmented image following the method described in section 3.5.1. Then, for each region class, a region classifier is trained with region features as input. We used our wrapper scheme detailed in section 3.5.2 to optimize the classifier performances. Three colour spaces are used in this experiment: RGB, HSV, and XYZ. The optimization of the SVM parameters increases the classifier performances of 5-10%. The best cross-validation rates are reached with q (quantization level) values superior to 50. We have also tested texture features but their performances are 10% inferior in mean than with the colour features as shown in Figure 20. Finally, the classifiers are trained a last time with the configurations giving the best cross-validation rates. The final set up of the different algorithms is then as follows (see Table 4):

Context	Seg. Algorithm (param.)	Class	Feature extractor param.		SVM param.	
			Colour space	q	C	γ
Context 1 (light green leaves)	CSC (41.9)	rose leaf	HSV	112	4	1
		white fly	HSV	112	1	4
Context 2 (dark green leaves)	CSC (48.7)	rose leaf	XYZ	21	64	4
		white fly	XYZ	21	256	0.25

Table 4. Set up of the segmentation, the feature extractors, and the classifiers.

5.5 Final segmentation quality assessment

In this section, we present the segmentation results on the test set. We compare six different methods, comprising (parts of) our approach and a pure top-down segmentation.

- method 1: *ad hoc* segmentation, with the Hysteresis thresholding algorithm tuned with $T_{\text{low}} = 0.45$ and $T_{\text{high}} = 1.0$,
- method 2: algorithm selection and tuning based on the learnt parameters from the whole training set (CSC is the best algorithm),
- method 3: method 2 + semantic segmentation (region labelling),
- method 4: algorithm selection and tuning based on image content analysis (one algorithm with learnt parameters per context),
- method 5: method 4 + semantic segmentation,
- method 6: over-segmentation + semantic segmentation.

The over-segmentation used in method 6 is performed with the CWAGM algorithm manually tuned with a very low region merging threshold (see Figure 20).

Performance scores of the test set are summarized in Table 5. Methods 3 and 5 give the best results. This outcome is predictable since the segmentation algorithm used for the method 5 is the same (CSC) and the parameter setting for the context 1 is close to the one for the context 2. The white fly region classifier for the context 2 has been trained on few samples since there are not many white flies on the front side of rose leaves. Consequently, the classification errors for the white fly class are higher for the method 5 context 2 than for the method 3. In a biological point of view, insects prefer to live hidden on the back side of the leaves, where they are better camouflaged (low contrast, not visible, etc.). Method 6 does not perform better results even if its initial over-segmentation is more precise (i.e. less missed boundary pixels) than with the CSC algorithm in methods 2 to 5.

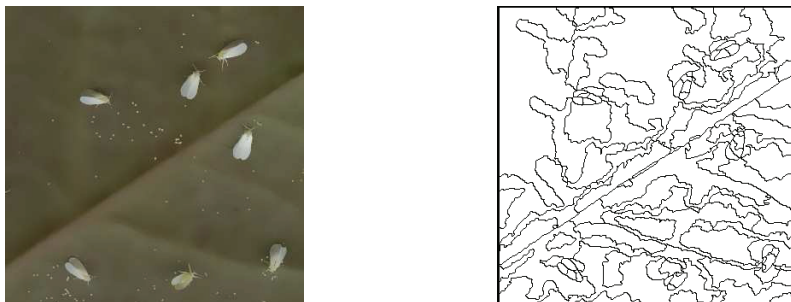


Fig. 20 Example of an initial over-segmented image used in method 6.

5.6 Evaluation on a public image database

In this section, we present evaluation results of the parameter optimization step on a public image database. The goal of the Berkeley Segmentation Dataset and Benchmark (BSD3) image database (Fowlkes & Martin, 2007) is to provide an empirical basis for research on image segmentation and boundary detection. To this end, the authors have collected 6000 hand-labelled segmentations of 300 Corel dataset colour images from 30 human subjects. The images depict natural scenes with at least one foreground object (e.g., an animal, a plant, a person, etc.). The ground truth are not labelled and the possible semantic classes are too numerous. Consequently, we do not assess the semantic segmentation part of our framework on this image database.

Method	Performance scores of segmentation of the test images			
	min	max	mean	std
1	0.00	0.35	0.09	0.08
2	0.00	0.78	0.21	0.16
3	0.00	0.65	0.12	0.14
4	0.06	0.83	0.23	0.17
5	0.06	0.62	0.12	0.14
6	0.00	0.67	0.15	0.14

Table 5. Statistics on the segmentation performances for the test set using different segmentation strategies.

The evaluation metric proposed in this image database for the benchmarking cannot be used with region-based segmentation algorithms since it relies on soft boundary maps of edge-based segmentation results (e.g. maps of gradient magnitude). We thus prefer our segmentation performance metric. For each image, several human segmentations exist (from three to eight) with different levels of refinements. We have decided to select the finest ones. Then, for each segmentation algorithm of our algorithm bank and for each image, algorithm parameters are optimized thanks to the selected manual segmentation. As previously done in section 5.2, we have compared the optimized segmentation achieved with the three optimization algorithm based on: the Simplex algorithm, the Genetic Algorithm, and a systematic search (see Table 6). Globally the three optimization algorithms perform in mean comparable results. This confirms the reliability of our parameter tuning approach for this image database.

Algorithm	$E_l^A = 0$ using Simplex / GA / Iterative search			
	min	max	mean	std
CSC	0.29 / 0.37 / 0.25	0.50 / 0.46 / 0.46	0.14 / 0.13 / 0.13	0.11 / 0.10 / 0.10
SRM	0.25 / 0.23 / 0.23	0.52 / 0.48 / 0.48	0.13 / 0.12 / 0.12	0.11 / 0.10 / 0.10
THRESH	0.19 / 0.19 / 0.38	0.35 / 0.35 / 0.35	0.11 / 0.11 / 0.11	0.09 / 0.09 / 0.09
EGBIS	0.21 / 0.20 / 0.34	0.73 / 0.71 / 0.71	0.37 / 0.37 / 0.39	0.14 / 0.14 / 0.14
CWAGM	0.22 / 0.22 / 0.50	0.44 / 0.44 / 0.46	0.12 / 0.12 / 0.19	0.09 / 0.09 / 0.08

Table 6. Statistics on the optimization performances for the training image set using the Simplex algorithm, the genetic algorithm and the systematic search.

7. Conclusion and future work

In this chapter, we address the problem of image segmentation with a cognitive vision approach. More precisely, we study three major issues of the segmentation task in vision systems: context adaptation, selection of an algorithm and tuning of its free parameters, according to the image content and to the application needs. Most of the time, this tedious and time-consuming task is achieved by an expert in image processing using a manual trial-and-error process. Recently, some attempts at automating the extraction of optimal parameters of segmentation have been made but they are still too application-dependent. The re-usability of such methods is still an open problem. We have chosen to handle this issue with a cognitive vision approach. Cognitive vision is a recent research field which proposes to enrich computer vision systems with cognitive capabilities, e.g., to reason from *a priori* knowledge, to learn from perceptual information, or to adapt its strategy to different problems.

We propose a supervised learning-based methodology for off-line configuration and on-line adaptation of the segmentation task in vision systems. The off-line configuration stage requires minimal knowledge to learn the optimal selection and tuning of segmentation algorithms. In an on-line stage, the learnt segmentation knowledge is used to perform an adaptive segmentation of images. This cognitive vision approach to image segmentation is thus a contribution for the research in cognitive vision. Indeed, it enables robustness, adaptation, and re-usability faculties to be fulfilled.

Finally, by addressing the problem of adaptive image segmentation, we have also addressed underlying problems, such as feature extraction and selection, and segmentation evaluation and mapping between low-level and high-level knowledge. Each of these well-known challenging problems is not easily tractable and still demands to be intensively considered. We have designed our approach (and our software) to be modular and upgradeable so as to take advantage of new progresses in these topics.

The brittleness of our approach to unknown situations is currently its major drawback. This concerns the context analysis level as well as the segmentation level. The concerned algorithms are the DBScan algorithm for image-content clustering and the SVMs for the semantic segmentation. Currently, neither the clustering algorithm nor the SVMs are able to adapt dynamically to new training data: the learning process must be run again on the whole training data set. The use of incremental machine learning techniques should be useful to fulfil the property of continuous learning. The main idea of incremental learning

for unforeseen situations is to dynamically adapt the clustering/classification method w.r.t. to the classification error of new input data. In our problem, unexpected situations can be identified thanks to the estimates of the context probability and the estimates of the SVM classification probabilities. The use of an adaptive classification algorithm using robust incremental clustering as proposed in (Prehn & Sommer, 2006) will then allow a dynamic update of the cluster and create new ones if necessary.

8. References

- Abdul-Karim, M.-A.; Roysam, B.; Dowell-Mesfin, N. M.; Jeromin, A.; Yuksel, M. & Kalyanaraman, S. (2005). Automatic selection of parameters for vessel/neurite segmentation algorithms, *Transactions on Image Processings*, Vol. 14(9), pp. 1338–1350
- Alvarado Moya, J. P. (2004). Segmentation of color images for interactive 3d object retrieval, *Ph.D. thesis*, Technical University of Aachen
- Bahnu, B.; Lee, S. & Das, S. (1995). Adaptive image segmentation using genetic and hybrid search methods, *Trans. on Aerospace and Electronic Systems*, Vol. 31(4), pp. 1268–1291
- Blum, A. L. & Langley, P. (1997). Selection of relevant features and examples in machine learning, *Artif. Intell.*, Vol. 97(1-2), pp. 245–271
- Borenstein, E. & Malik, J. (2006). Shape guided object segmentation, in: *Proc. of the Int. Conf. on Computer Vision and Pattern Recognition*, Vol. 1, pp. 969–976
- Burges, C. J. C. (1998). A tutorial on support vector machines for pattern recognition, *Data Mining and Knowledge Discovery*, Vol. 2(2), pp. 121–167
- Chen, Y. & Wang, J. Z. (2004). Image categorization by learning and reasoning with regions, *Journal of Machine Learning Research*, Vol. 5, pp. 913–939
- Cinque, L.; Corzani, F.; Levialdi, S.; Cucchiara, R. & Pignatelli, G. (2002). Improvement in range segmentation parameters tuning, in: *Proc. of the Int. Conf. on Pattern Recognition*, Vol. 1, pp. 10176
- ECVISION (2005). A research roadmap of cognitive vision, *Technical report*, Project IST-2001-35454
- Ester, M.; Kriegel, H.-P.; Sander, J. & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise, in: *Proc. of the Int. Conf. on Knowledge Discovery and Data Mining*, Portland, pp. 226–231
- Everingham, M.; Muller, H. & Thomas, B. (2002). Evaluating image segmentation algorithms using the pareto front, in: *Proc. of the Eur. Conf. Computer Vision*, Vol. 2353(4), pp. 34–38
- Felzenszwalb, P. F. & Huttenlocher, D. P. (2004). Efficient graph-based image segmentation, *Int. Journal on Computer Vision*, Vol. 59(2), pp. 167–181
- Fowlkes, C. & Martin, D. (2007). The berkeley segmentation dataset and benchmark, <http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/>
- Gelasca, E.; Salvador, E. & Ebrahimi, T. (2003). Intuitive strategy for parameter setting in video segmentation, in: *Visual Communications and Image Processing*, Vol. 5150, pp. 998–1008
- Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Longman Publishing Co., Boston, MA, USA

- Guyon, I. & Elisseeff, A. (2003). An introduction to variable and feature selection, *J. Mach. Learn. Res.*, Vol. 3, pp. 1157–1182
- Haralick, R. (1979). Statistical and structural approaches to texture, in: *Proc. of the IEEE*, Vol. 67, pp. 67
- Huang, F. J. & LeCun, Y. (2006). Large-scale learning with svm and convolutional nets for generic object categorization, in: *Proc. of the Int. Conf. on Computer Vision and Pattern Recognition*, pp. 284–291
- Jain, A. K. & Farrokhnia, F. (1991). Unsupervised texture segmentation using gabor filters, *Pattern Recognition*, Vol. 24(12), pp. 1167–1186
- Kohavi, R. & John, G. H. (1997). Wrappers for feature subset selection, *Artificial Intelligence*, Vol. 97(1-2), pp. 273–324
- Laws, K. (1980). Textured image segmentation, *Ph.D. thesis*, Univ. Southern California
- Lucchese, L. & Mitra, S. (2001). Color image segmentation : A state-of-the-art survey, Vol. 67, *Indian National Science Academy*, New Dehli, pp. 207–221
- Mao, S. & Kanungo, T. (2000). Automatic training of page segmentation algorithms : An optimizatin approach, in: *Proc. of the Int. Conf. on Pattern Recognition*, Barcelona, Spain, pp. 531–534
- Maurer, C. R. J.; Qi, R. & Raghavan, V. (2003). A linear time algorithm for computing exact euclidean distance transforms of binary images in arbitrary dimensions, *Trans. on Pattern Anlasys and Machine Intelligence.*, Vol. 25(2), pp. 265–270
- Mezaris, V.; Kompatsiaris, I. & Strintzis, M. (2003). Still image objective segmentation evaluation using ground truth, in: *Proc. of the Workshop on Information and Knowledge Management for Integrated Media Communication*, Prague, Czech Republic, pp. 9–14
- Molina, L. C.; Belanche, L. & Nebot, A. (2002). Feature selection algorithms : A survey and experimental evaluation, in: *Proc. of the Int. Conf. on Data Mining*, Japan, pp. 306–313
- Nelder, J. & Mead, R. (1965). A simplex method for function minimization, *Computer Journal*, Vol. 15, pp. 1162–1173
- Nock, R. & Nielsen, F. (2004). Statistical region merging, *Pattern Analysis and Machine Intelligence*, Vol. 26(11), pp. 1452–1458
- Pal, N. R. & Pal, S. K. (1993). A review on image segmentation techniques, *Pattern Recognition*, Vol. 26(9), pp. 1277–1294
- Peng, J. & Bahnu, B. (1998). Delayed reinforcement learning for adaptive image segmentation and feature extraction, *Systems, Man, and Cybernetics*, Vol. 28(3), pp. 482–488
- Pignalberi, G.; Cucchiara, R.; Cinque, L. & Levialdi, S. (2003). Tuning range image segmentation by genetic algorithm, *EURASIP Journal on Applied Signal Processing*, Vol. (8), pp. 780–790
- Prehn, H. & Sommer, G. (2006). An adaptive classification algorithm using robust incremental clustering, in: *Proc. of the Int. Conf. on Pattern Recognition*, pp. 896–899
- Priese, V.; Rehrmann, L. & Sturm, P. (2002). Color structure code. URL <http://www.uni-koblenz.de/>
- Reed, T. R. & du Buf, J. M. H. (1993). A review of recent texture segmentation and feature extraction techniques, *CVGIP: Image Underst.*, Vol. 57(3), pp. 359–372
- Rosenberger, C.; Chabrier, S.; Laurent, H. & Emile, B. (2005). Unsupervised and Supervised Segmentation evaluation, in book: *Advances in Image and Video Segmentation*, Yu-Jin Zhang editor, Tsinghua University, Beijing, China, chapter XVIII

- Schnitman, Y.; Caspi, Y.; Cohen-Or, D. & Lischinski, D. (2006). Inducing semantic segmentation from an example, in: *ACCV*, Vol. 3852, Springer-Verlag, pp. 393–384
- Skarbek, W. & Koschan, A. (1994). Colour image sementation - a survey, *Technical report*, Technical University of Berlin
- Wu, T.; Lin, C. & Weng, R. (2004). Probability estimates for multi-class classification by pairwise coupling, *The Journal of Machine Learning Research*, Vol. 5, pp. 975–1005
- Xia, Y.; Feng, D.; Rongchun, Z. & Petrou, M. (2005). Learning-based algorithm selection for image segmentation, *Pattern Recognition Letters*, Vol. 26(8), pp. 1059–1068
- Yasnoff, W.; Mui, J. & Bacus, J. (1977). Error measures for scene segmentation, *Pattern Recognition*, Vol. 9, pp. 217–231
- Zhang, Y. (1996). A survey on evaluation methods for image segmentation, *Pattern Recognition*, Vol. 29(8), pp. 1335–1346
- Zhang, Y. & Luo, H. (2000). Optimal selection of segmentation algorithms based on performance evaluation, *Optical Engineering*, Vol. 39, pp. 1450–1456
- Zhang, H.; Fritts, J. E. & Goldman, S. A. (2008). Image segmentation evaluation: A survey of unsupervised methods, *Computer Vision and Image Understanding*, Vol. 110(2), pp. 260–280