Available online at www.sciencedirect.com
SCIENCE DIRECT®
Pattern Recognition 38 (2005) 2099–2113
PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY
www.elsevier.com/locate/patcog

Inexact graph matching for model-based recognition: Evaluation and comparison of optimization algorithms

Roberto M. Cesar Jr.[a], Endika Bengoetxea[b], Isabelle Bloch[c,*], Pedro Larrañaga[b]

[a]*Department of Computer Science, IME, University of São Paulo, São Paulo, Brazil*
[b]*Department of Computer Architecture and Technology, University of the Basque Country, San Sebastian, Spain*
[c]*Department TSI, CNRS UMR 5141 LTCI, GET, ENST, Paris, France*

Received 9 August 2004; received in revised form 16 May 2005; accepted 16 May 2005

Abstract

A method for segmentation and recognition of image structures based on graph homomorphisms is presented in this paper. It is a model-based recognition method where the input image is over-segmented and the obtained regions are represented by an attributed relational graph (ARG). This graph is then matched against a model graph thus accomplishing the model-based recognition task. This type of problem calls for inexact graph matching through a homomorphism between the graphs since no bijective correspondence can be expected, because of the over-segmentation of the image with respect to the model. The search for the best homomorphism is carried out by optimizing an objective function based on similarities between object and relational attributes defined on the graphs. The following optimization procedures are compared and discussed: deterministic tree search, for which new algorithms are detailed, genetic algorithms and estimation of distribution algorithms. In order to assess the performance of these algorithms using real data, experimental results on supervised classification of facial features using face images from public databases are presented.

Keywords: Inexact graph matching; Graph homomorphism; Tree search; Estimation of distribution algorithms; Model-based structure recognition

1. Introduction

In this paper we propose a method for segmentation and recognition of image structures or objects based on a model of the imaged scene. The idea is to represent knowledge about the structures in the model as a graph. Based on an over-segmentation, input (i.e. target) image information is also represented as a graph. More specifically, an *attributed relational graph* (ARG) is used to represent the model and the image content. Therefore, the recognition procedure amounts to find a suitable matching between both graphs.

Graph representations are widely used for dealing with structural information in different domains such as networks, psycho-sociology, image interpretation and pattern recognition, to name but a few. One important problem to be solved when using such representations is graph matching. In order to achieve a good correspondence between two graphs, the most used concept is the graph isomorphism and a lot of work is dedicated to the search for the best isomorphism between two graphs or subgraphs [1]. However in a number of cases, the bijective condition is too strong, and the problem is expressed rather as an inexact graph matching

* Corresponding author. Tel.: 33 1 45 81 75 85; fax: 33 1 45 81 37 94.
E-mail addresses: cesar@ime.usp.br (R.M. Cesar), endika@si.ehu.es (E. Bengoetxea), Isabelle.Bloch@enst.fr (I. Bloch), ccplamup@si.ehu.es (P. Larrañaga).

0031-3203/$30.00
doi:10.1016/j.patcog.2005.05.007

problem. For instance, inexact graph matching appears as an important area of research in the pattern recognition field. In several approaches graphs are used to represent knowledge and information extracted from images, where vertices represent the regions or entities of the image and edges show the relationships between them. Cartography, robotics and autonomous agents, character recognition, and recognition of brain structures are examples of areas in which this type of representation appears. Because of the usually schematic aspect of the model and difficulty to accurately segment the image into meaningful entities, no isomorphism can be expected between both graphs. Such problems call for inexact graph matching. The technique of inexact graph matching has been extensively studied in several different domains such as pattern recognition, computer vision, cybernetics, among others [2]. Most works on inexact graph matching rely on the optimization of some objective function. This function usually measures the adequacy between vertices and between edges of both graphs, involving both the similarity between attributes of vertices and the similarity between attributes of edges, as well as the structure of the graph. Existing optimization methods include combinatorial optimization techniques [3], relaxation techniques [4,5], expectation maximization (EM) algorithm [6,7], estimation of distribution algorithms (EDAs) [8], and genetic algorithms [9]. Other methods are more concerned by the structure itself of the graphs and use tree search and propagation techniques [10], heuristic-based graph traversing [11], graph editing [12,13] and graph labeling based on probabilistic models of attributes [14].

In a previous paper [15], estimation of distributions algorithms have been developed for inexact graph matching. Here we develop new algorithms based on tree search, which constitute one of the contribution of the paper. As a result, the following optimization procedures are applied to real-world data, and then compared and discussed in this paper: deterministic tree search, classical genetic algorithms and estimation of distribution algorithms. Another main contribution of this paper is to adapt them to the problem of model-based recognition problem through matching optimization, and to compare and evaluate them on real data.

An experiment about the recognition of facial features has been devised in order to evaluate the aforementioned optimization algorithms used in our method. Face recognition has received intense and growing attention from the computer vision community, partially because of the many applications such as human-computer interaction, model-based coding, teleconferencing, security and surveillance. A particularly important task that arises in different problems of face recognition is the location and segmentation of facial feature regions, such as eyebrows, iris, lips, and nostrils [16]. For instance, segmentation of facial regions is important for feature-based face recognition [17], medical applications [18] and analysis of facial expressions [16]. In this paper, we have applied our method to segment and recognize facial features using face images from public databases, with the aim of comparing the optimization algorithms.

This paper is organized as follows. In Section 2, we present our graph-based approach for solving model-based recognition of image structures. The optimization algorithms assessed in this paper are discussed in Section 3. The experimental results evaluating and comparing these algorithms are described in Section 4. Some comments on our ongoing research are given in Section 5.

2. Homomorphism between ARGs

2.1. Notations and definitions

In this work, $\tilde{G} = (N, E)$ denotes a directed graph where N represents the set of vertices of $\tilde{G}$ and $E \subseteq N \times N$ the set of edges. Two vertices a, b of N are adjacent if $(a, b) \in E$. If each vertex of $\tilde{G}$ is adjacent to all others, then $\tilde{G}$ is said to be complete. We define an attributed relational graph as $G = (N, E, \mu, \nu)$, where $\mu : N \to L_N$ assigns an attribute vector to each vertex of N. Similarly, $\nu : E \to L_E$ assigns an attribute vector to each edge of E. We typically have $L_N = \mathbb{R}^m$ and $L_E = \mathbb{R}^n$, where m and n are the numbers of vertex and edge attributes, respectively. ARGs have been extensively used in computer vision and artificial intelligence in problems of model-based recognition and structural scene analysis. In such approaches, the scene is composed of a number of objects arranged according to some structure. Usually, each object is represented by a vertex of the ARG, being characterized by a set of features encoded by the vertex attribute vector, while the ARG edges represent the relation between objects. The vertices and the edges attributes are called object and relational attributes, respectively. In the present approach, we need two ARGs $G_1 = (N_1, E_1, \mu_1, \nu_1)$ and $G_2 = (N_2, E_2, \mu_2, \nu_2)$, which will be henceforth referred to as the *input* (i.e. derived from the image) and the *model* graphs, respectively. $|N_1|$ denotes the number of vertices in N_1, while $|E_1|$ denotes the number of edges in E_1. We use a subscript to denote the corresponding graph, e.g. $a_1 \in N_1$ denotes a vertex of G_1, while $(a_1, b_1) \in E_1$ denotes an edge of G_1. We use a superscript to enumerate the nodes of a graph, i.e. $a_1^1, a_1^2, \ldots, a_1^{|N_1|} \in N_1$. Similar notations are used for G_2.

The easiest way of obtaining the model graph G_2 is by manually segmenting a prototype image followed by representing it by the corresponding ARG. Section 4 illustrates this by showing the generated model for our experiments. As far as G_1 is concerned, the input image is segmented by applying a watershed algorithm to the image gradient, which results in an over-segmented image (any other over-segmentation method could be used as well). The input graph G_1 is subsequently obtained from the over-segmented image, with each vertex representing an image connected region. Each vertex of G_1 is adjacent to all other G_1 vertices,

and it is also adjacent to itself, i.e. $\forall a_1 \in N_1, (a_1, a_1) \in E_1$. The same applies to G_2.

The *association graph* $\tilde{G}_A$ between G_1 and G_2 is defined as the complete graph $\tilde{G}_A = (N_A, E_A)$, with $N_A = N_1 \times N_2$.

A *graph homomorphism* h between G_1 and G_2 is a mapping $h : N_1 \to N_2 | \forall a_1 \in N_1, \forall b_1 \in N_1$, if $(a_1, b_1) \in E_1$, then $(h(a_1), h(b_1)) \in E_2$. Note that $|N_1|$ is often much greater than $|N_2|$ in model-based recognition problems where each object of the image can be subdivided in several regions (over-segmentation).

A solution of an inexact match problem between G_1 and G_2 can be expressed as a subgraph $\tilde{G}_S = (N_S, E_S)$ of the association graph $\tilde{G}_A$ between G_1 and G_2 with $N_S = \{(a_1, a_2), a_1 \in N_1, a_2 \in N_2\}$ such that $\forall a_1 \in N_1, \exists a_2 \in N_2, (a_1, a_2) \in N_S$ and $\forall (a_1, a_2) \in N_S, \forall (a'_1, a'_2) \in N_S,\ a_1 = a'_1 \Rightarrow a_2 = a'_2$ which guarantees that each vertex of the image graph has exactly one label (vertex of the model graph) and $|N_S| = |N_1|$. Such solution defines a homomorphism between G_1 and G_2. $\tilde{G}_S$ is built as a clique of $\tilde{G}_A$. This approach only considers the structure of the graphs.

Clearly, there are many possible homomorphisms that represent an inexact match between G_1 and G_2, and we need to define an objective function to assess the quality of a given homomorphism and its suitability with respect to each specific application. This criterion should include, additionally to the structural aspects, information on the attributes. In particular, the homomorphism should minimize the dissimilarity between the object attributes of the mapped vertices from G_1 to G_2 and the respective relational attributes associated to the matched edges.

2.2. Objective function

The evaluation of the quality of a solution expressed by $\tilde{G}_S$ is performed through an objective function. Finding the best solution then amounts to minimize this function. In this paper we have explored the following objective function:

$$f_1(\tilde{G}_S) = \frac{\alpha}{|N_S|} \sum_{(a_1,a_2)\in N_S} c_N(a_1, a_2) + \frac{(1-\alpha)}{|E_S|} \sum_{e \in E_S} c_E(e), \quad (1)$$

where $c_N(a_1, a_2)$ is a measure of the adequacy between a_1 and a_2, i.e. a measure of dissimilarity (or similarity if the objective function is to be maximized) between the attributes of a_1 and a_2. Similarly, if $e = ((a_1, a_2), (b_1, b_2))$, $c_E(e)$ is a measure of the dissimilarity between edge (a_1, b_1) of the image and edge (a_2, b_2) of the model. Note that c_N and c_E are supposed to be normalized between 0 and 1 in Eq. (1). Typically, $c_N(a_1, a_2)$ will be defined as a decreasing function of the similarity between vertex attributes. If two vertices a_1 and a_2 have the same attributes (high similarity), then c_N will be very low and the association of a_1 and a_2 will be favored when minimizing f_1. On the other hand, associations between nodes having different attribute values will be penalized. The term depending on edge comparison can be interpreted in a similar way. The specific form of the similarity or dissimilarity measures depends on the type of attributes. Some examples will be given below for the application chosen to illustrate the approach.

The function f_1 is a simple weighted average of measures of qualities of vertex associations (first sum) and of edge associations (second sum). Weighted mean operators achieve a compromise between the similarities. Such operators have a compensation effect which is interesting in cases where both low and high similarities occur. For instance a low similarity between a model node (or edge) and an image node (or edge) with respect to one attribute can be compensated by a high similarity with respect to another attribute. Such operators allow to weight differently node attributes and edge attributes, or to give more importance to some attributes than to other ones. This is particularly useful when characteristics of objects or of relations have not the same level of stability and variability. For instance if an attribute corresponds to a highly variable feature for which differences are expected between the model and the image, a higher dissimilarity can be expected between the model and the image, but it should have a low impact on the global objective function. One noticeable point in f_1 is that it only considers the quality of the actual associations. It does not take into account possible similarities between vertices or edges that have not been matched. Nevertheless, solutions where a lot of non-matched nodes have a good similarity should not be favored. In order to take this into account, a more sophisticated function could be defined.

3. The optimization algorithms

Among the different possible approaches for finding a suitable homomorphism based on the objective function described in the previous section, we developed, tested and compared several methods based on heuristics and on evolutionary computation techniques. We already applied some of them to other applications [9,19,20].

We explore a beam search algorithm often applied to different problems in artificial intelligence. Here we propose a new algorithm of this type adapted to inexact graph matching (a preliminary version was shortly introduced in Ref. [21]). Evolutionary computation techniques are proposed in order to reduce the number of solutions to be analyzed. The evolutionary computation algorithms that we have applied and tested are Genetic Algorithms (GAs) and Estimation of Distribution Algorithms (EDAs), which are also discussed below and adapted to the aim of this work.

3.1. Tree search algorithm

This algorithm finds a solution by creating a search tree with each vertex representing a pair (k, l), where k represents the kth vertex of the input graph (i.e. a_1^k) and l represents the lth vertex of the model graph (i.e. a_2^l), thus analogous to the nodes of the association graph $\tilde{G}_S$ introduced in Section 2.1. The search tree is initialized with a dummy root vertex $(0, 0)$ that is expanded in $|N_2|$ sons $(1, 1), (1, 2), \ldots, (1, |N_2|)$. The chosen objective function f_1 (see Section 2.2) is calculated for each son. The cheaper leaf in the tree is taken to be expanded in the next loop. In this first expansion of the tree, the only leaves are the nodes that have just been expanded, but this will not be the case after the second node expansion. It is hence necessary to calculate $c_N(a_1^1, a_2^j)$, $j = 1, \ldots, |N_2|$ (see Eq. (1)), i.e. the cost of matching the input graph node a_1^1 to each model graph node a_2^j. This first step does not involve calculating the edge costs c_E since only one node of each graph is being considered so far. Suppose that $(1, 3)$ is chosen. i.e. a_1^1 is matched to a_2^3. Then, $(1, 3)$ is analogously expanded in $|N_2|$ sons $(2, 1), (2, 2), \ldots, (2, |N_2|)$, the objective function for each newly born son is calculated and the cheaper tree leaf among all leaves (including nodes left unexpanded in previous steps) is taken to be expanded. It is necessary to calculate $c_N(a_1^2, a_2^j)$, as well as the edge costs$c_E((a_1^1, a_1^2), (a_2^3, a_2^j))$, $j = 1, \ldots, |N_2|$. As new nodes are expanded more terms c_N and c_E are taken into account by the objective function. All matchings between edges must be taken into account once a tree leaf is expanded. For instance, suppose that $(2, 4)$ is chosen as the cheapest node. Then $(2, 4)$ is expanded in $|N_2|$ sons $(3, 1), (3, 2), \ldots, (3, |N_2|)$. In order to calculate the value of the objective function for each new node, it is necessary to calculate $c_N(a_1^3, a_2^j)$, as well as the edge costs $c_E((a_1^2, a_1^3), (a_2^4, a_2^j))$, $j = 1, \ldots, |N_2|$, and $c_E((a_1^1, a_1^3), (a_2^3, a_2^j))$, $j = 1, \ldots, |N_2|$. The normalization terms $|N_A|$ and $|E_A|$ must be set properly when calculating the objective function for each node since the number of considered vertices and arcs depends on the depth of the exploded nodes.

It is worth noting that all leaves are considered at each step, i.e. tree nodes previously left unexpanded are also candidates to be expanded. For instance, in our previous example, at the third iteration we had:

Expanded nodes so far: $(1, 3)$ and $(2, 4)$.

Nodes to be considered for being expanded:$(1, j)$, $j = 1, \ldots, |N_2|$; $j \neq 3$ and $(2, j)$, $j = 1, \ldots, |N_2|$; $j \neq 4$.

The process is repeated until a tree vertex $(|N_1|, l)$ is reached, meaning that all $|N_1|$ vertices in G_1 have been assigned to a vertex in G_2, thus defining a suitable homomorphism between the two graphs. This procedure is summarized in Algorithm 1.

Algorithm 1 defines a priority queue and returns a pointer to the cheaper vertex when the $p = priority\_remove(pl)$ is called. The cost of each tree vertex is calculated within the *explode* function based on the adopted objective function and involves the evaluation of $c_N(a_1, a_2)$ and $c_E(e)$ for the corresponding vertices and edges.

Algorithm 1 Tree search algorithm

```
pl = priority_init(); /* initializes the priority queue */
p0 = tree_init(); /* initializes the search tree */
p = p0;
priority_insert(pl, p); /* insert a tree pointer in the
   priority queue */
endflag = 0;
while (endflag ≠ 1) do
  p = priority_remove(pl); /* p is the cheaper vertex */
  if (solution(p, input_graph)) then
    improve_solution(p);
    save_solution(fpout,p);
    endflag = 1;
  else
    explode(p, input_graph, model_graph); /* expands
      the cheaper vertex */
  end if
  priority_insert_sons(pl, p); /* insert the newly born
     leaves in the priority queue */
end while
```

We have limited the maximum size allowed for the priority queue and, once this limit is reached, the more expensive vertices are discarded from the queue. This solution is similar to the beam search algorithm, which saves time and space complexity at the cost of not considering many paths in the tree (and therefore possibly loosing a better solution). We found in practice that this limitation allows the algorithm to converge fastly and that the solution quality does not critically depend on the maximum allowed size.

One of the main drawbacks with this approach is that a solution is created based on "local" decisions, i.e. each node is labeled by choosing the cheapest tree leaf from the priority queue. Clearly, the node price at each step only takes into account the tree nodes above the leaf under consideration, and the remaining subtree is not considered (because it has not been expanded yet), which often leads to suboptimal solutions (i.e. local minima). We have proposed a randomized variation of the tree search method and investigated the solutions produced by it in Ref. [22]. The idea is to randomize the search by creating a priority list of k candidates ($k = 100$ in our experiments) having the smallest dissimilarity measures. Then, the next leaf to be expanded is chosen randomly among the candidates of this list. This procedure is executed several times, and since different executions may lead to different solutions, we select the best one. This random algorithm provides much better results than the deterministic one in many runs [22], thus suggesting that the tree-search algorithm could be improved.

Therefore, we have implemented a post-processing step in order to get the solution subsequently improved. Once a solution is reached by the above procedure, the algorithm tracks its path in the tree, from leaf to root, and verifies the price of the solution obtained by changing the node label by the other possible labels for that node. If the obtained solution is cheaper than the previous one, then it is updated with the new label for that node (otherwise, nothing is done). The algorithm then proceeds for the next node in the tree. If the solution has been improved at least once after traversing a leaf-to-root path, this procedure is repeated again from the leaf. Convergence is reached after traversing the leaf-to-root path with no improvements on the solution. This final procedure is represented as the *improve_solution*(p) step in algorithm 1.

3.2. Genetic algorithms

GAs [23,24] are stochastic heuristic evolutionary computation techniques. They keep a population formed by individuals (i.e. solutions), and make it evolve towards better solutions according to a fitness function by selecting individuals and applying cross-over and mutation operators to them. In our problem, each individual represents a correspondence hypothesis, that is, a correspondence for each of the vertices in the input graph to a vertex in the model graph.

The individuals in the GA represent a solution for the optimization problem stated in the previous section. In our concrete case, we propose to use an individual representation which contains $|N_1|$ genes (one per each node of G_1 that has to be recognized), and in which each gene can contain an integer value between 1 and $|N_2|$ (that is, a label representing a node of the model graph G_2). More formally, given an individual $\boldsymbol{x} = (x_1, \ldots, x_{|N_1|})$, the fact that $x_i = k$, for $1 \leqslant i \leqslant |N_1|$ and $1 \leqslant k \leqslant |N_2|$, means that this solution proposes to match ith vertex of G_1 with the kth vertex of G_2. The fitness value of the solution $\boldsymbol{x}$ is computed using the objective function (Section 2.2) as fitness function.

Regarding the literature on genetic algorithms, we have chosen two types of general purpose GAs. The first one is known as the elitist genetic algorithm (eGA) [25], and consists in keeping the best individual in a population and regenerate all the other individuals for the next generation.

The second GA type applied is steady-state (ssGA) [26], where two individuals are randomly chosen in a generation, undergo cross-over in order to generate a new individual, and this is then compared to the worst individual of the generation. The worst of both is then removed.

The eGA was programmed to stop the search automatically at a maximum of 100 generations. In the case of ssGA, as it generates only a new individual each iteration, it was programmed in order to generate the same number of individuals as in eGA.

The initial population for both GAs was generated using a random generation procedure based on a uniform distribution for all the possible values. In both GA types, a population size of 2000 individuals was chosen, with a mutation probability of $1.0/|N_1|$ (where $|N_1|$ is the number of vertices of the input graph) and a cross-over probability of 1. We decided to set this last parameter to its maximum value since all the different genetic algorithms tested in our experiments are elitist ones, and therefore we ensure that the fittest individual of each generation will remain in the next one. The reason for using an elitist approach in all our genetic algorithms is that the theoretical proof of GAs to converge to the best solution can be done for this type of GAs. Furthermore, usually eGAs obtain better results than canonical GAs, and after our preliminary tests we concluded that this fact was also true for our type of problems.

3.3. Estimation of distribution algorithms

EDAs [8,27], which were introduced about 10 years ago [28,29], are also stochastic heuristic search strategies within the evolutionary computation approaches, where, similarly as GAs, a number of solutions or individuals are created every generation, thus evolving until a satisfactory solution is achieved.

The motivation for the use of EDAs is that the behavior of GAs depends to a large extent on the choice of operators and probabilities for crossing and mutation, size of the population, rate of generational reproduction, the number of generations, and others. Experience on the use of these algorithms is required in order to choose the suitable values for these parameters. In addition, EDAs have already shown their better performance in many classical optimization problems [8] and also than other inexact graph matching techniques [30].

The main difference between EDAs and other evolutionary search strategies is that the evolution of the population from a generation to the next one is performed by estimating the probability distribution of the fitter individuals, creating a probabilistic graphical model. If the individuals are composed of discrete values, this probabilistic graphical model has the form of a Bayesian network. The next generation of fitter individuals is created by sampling the induced model. Therefore, in EDAs the new population of individuals is generated without using neither cross-over nor mutation operators. Instead, the new individuals are sampled starting from the probability distribution estimated from the database containing only selected individuals from the previous generation.

The representation of solutions (i.e. individuals) and the fitness value of the solution for EDAs have been chosen exactly the same as for GAs.

Let $\boldsymbol{X} = (X_1, \ldots, X_n)$ be a set of discrete random variables, and let x_i be a value of X_i, the ith component of $\boldsymbol{X}$. Let $\boldsymbol{y} = (x_i)_{X_i \in \boldsymbol{Y}}$ be a value of $\boldsymbol{Y} \subseteq \boldsymbol{X}$. A probabilistic graphical model for $\boldsymbol{X}$ is a graphical factorization of the joint probability distribution, $p(\boldsymbol{X} = \boldsymbol{x})$ values. The representation

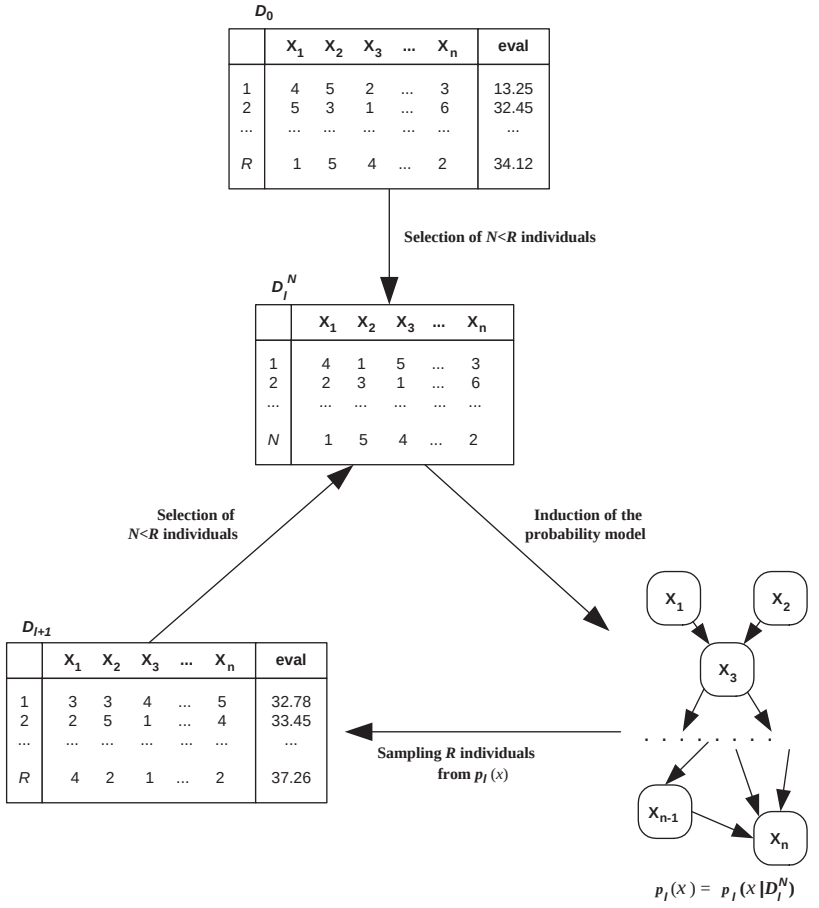

Fig. 1. Illustration of the EDA approach in optimization.

of this model is given by two components: a structure S and a set of conditional probabilities.

The structure S of the model for $\boldsymbol{X}$ is a directed acyclic graph (DAG) that describes a set of conditional interdependencies between the variables on $\boldsymbol{X}$. Let $\boldsymbol{Pa}_i^S$ represent the set of parents-variables from which an arrow is coming out in S- of the variable X_i in the probabilistic graphical model. The structure S for $\boldsymbol{X}$ assumes that X_i and its non descendants are independent given $\boldsymbol{Pa}_i^S$, $i = 2, \ldots, n$. Therefore, the factorization can be written as follows:

$$p(\boldsymbol{x}) = p(x_1, \ldots, x_n) = \prod_{i=1}^{n} p(x_i \mid \boldsymbol{pa}_i^S). \tag{2}$$

The EDA approach is illustrated in Fig. 1. Four main steps are identified in all EDAs:

(1) Firstly, the initial population D_0 of M individuals is generated. The generation of these M individuals is usually carried out by assuming a uniform distribution on each variable, and next each individual is evaluated. This has also been done in our EDAs ($M = 2000$ in our experiments).

(2) Secondly, in order to make the $(l-1)$th population D_{l-1} evolve towards the next one (D_l), a number N ($N \leqslant M$) of individuals are selected according to a criterion. We denote by D_{l-1}^{Se} the set of selected individuals from generation $(l-1)$. In our case, we have decided to choose the half of the population ($N = M/2$) formed by the N fittest individuals.

(3) Thirdly, the n-dimensional probabilistic model $p_l(\boldsymbol{x}) = p(\boldsymbol{x}|D_{l-1}^{Se})$ that better represents the inter-dependencies between the n variables is induced. This step is also known as the *learning* procedure, and it is the most crucial one, since representing appropriately the dependencies between the variables is essential for a proper evolution towards fitter individuals.

(4) Finally, the new population D_l constituted by M new individuals is obtained by carrying out the simulation of the probability distribution learned in the previous step. The simulation method used in our experiments is the one known as *Probabilistic Logic Sampling* (PLS), which was proposed in Ref. [31]. Usually an elitist approach is followed, and therefore the best individual of population D_{l-1} is kept in D_l. As a result, a total of

$M-1$ new individuals is created every generation instead of M.

Steps 2–4 are repeated until a stopping condition is verified. In our particular case, we have chosen to stop the search when uniformity in the generated population is obtained, or when a maximum of 100 generations has been reached.

A common way of classifying the different EDAs in the literature is by taking into account the maximum number of inter-dependencies between variables that they accept (maximum number of parents that a variable X_i can have in the probabilistic graphical model). The reader can find in Ref. [8] a more complete review of this topic. Usually, 3 main categories are identified:

(1) Without interdependencies: All the EDAs belonging to this category assume that the n-dimensional joint probability distribution factorizes as a product of n univariate and independent probability distributions, that is $p_l(\boldsymbol{x}) = \prod_{i=1}^{n} p_l(x_i)$.
An illustrative example in this category is the Univariate Marginal Distribution Algorithm (UMDA) [32], in which the relative marginal frequencies of the ith variable within the subset of selected individuals D_{l-1}^{Se} are estimated.
(2) Pairwise dependencies: all the EDAs in this category estimate the joint probability distribution by only taking into account dependencies between pairs of variables. An example of this second category is the Mutual Information Maximization for Input Clustering (MIMIC) proposed in Refs. [33,34].
(3) Multiple interdependencies: Estimation of Bayesian Networks Algorithm [35] (EBNA) is an example of an EDA that belongs to this category. EBNA takes into account multiple inter-dependencies between variables by constructing a probabilistic graphical model with no restriction in the number of parents that each of the variables can have. Another example of an EDA in this category is the Factorised Distribution Algorithm (FDA) [27], which is based on a fixed probabilistic graphical model and therefore it does not add a learning step as in former EBNA.
The three different EDAs mentioned so far (UMDA, MIMIC, and EBNA) have been applied in this paper, each of them as representatives of their corresponding EDA category.

4. Experimental results

4.1. Application: facial feature recognition

The proposed approach has been applied in a series of experiments for facial feature segmentation based on finding a suitable homomorphism between two ARGs. First the facial landmarks are located by a tracking method and used to define a limited region that is considered by the recognition procedure (see Fig. 2). The model is manually obtained from a reference image. The image in which recognition has to be performed is segmented using a watershed method which provides an over-segmentation (see Fig. 3). Both input and

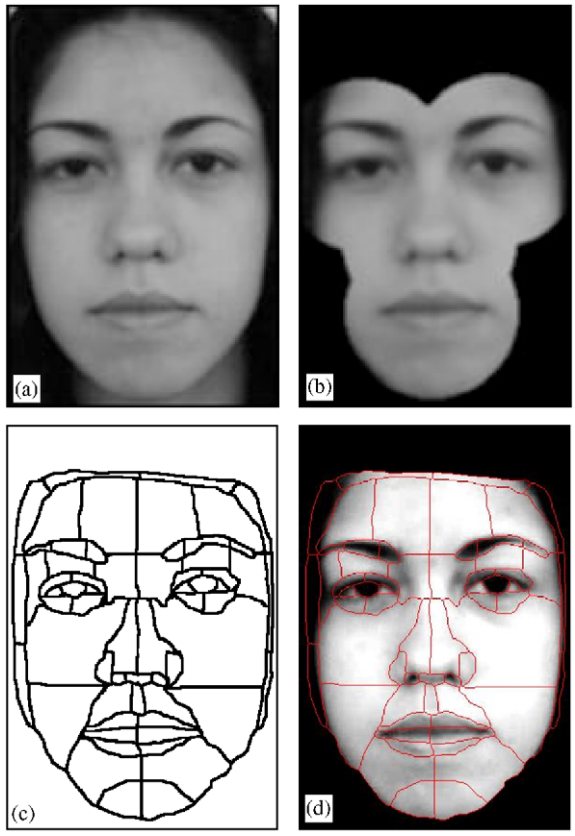

Fig. 2. (a) Original image; (b) its masked version containing only the regions of interest around the landmarks; (c) face model manually segmented; (d) model superimposed to the face image.

Fig. 3. Over-segmented image by the watershed algorithm.

model faces are represented by ARGs and the respective attributes are computed. The objective function is optimized using one of the three proposed methods and the final segmentation produced by the method may be evaluated.

The initial step to segment the facial feature regions is to locate the face in the image, which can be done both in still images and in video sequences. The latter means to detect the face in a frame and to track it in the subsequent frames. In our approach, these steps are performed by a recently proposed technique that represents a face using the Gabor wavelet network (GWN) [36]. Furthermore, the GWN acts as a rigid model that provides *approximative landmarks* which are located near the facial features to be segmented [37] (e.g. eyes, nose and mouth). These landmarks are used in two different ways in order to make our approach more efficient: (1) only certain regions around the landmarks are considered; and (2) the landmark information is used by the optimization algorithm to constrain the solution search space. The GWN allows a real-time efficient face tracking based on the whole face, being robust to facial feature deformations such as eye blinking and smile.[1] The reader is referred to Ref. [37] for complete details about the GWN approach.

The model graph should contain vertices associated to each facial feature of interest, e.g. for each eyebrow, iris, nostril, mouth and the skin. It is important to note that, in the model of Fig. 2(c), some single facial features have been subdivided, e.g. the eyebrows. This has been done because the adopted vertex and edge attributes are calculated based on average measures over the segmented image regions. Therefore, model attributes extracted from large regions tend to be less representative because such regions often present larger variability with respect to the attributes. Some facial features have thus been subdivided in order to circumvent such a potential problem. Furthermore, the fact that the skin is not a well-localized facial feature (in contrary to pupils and nostrils) presents an additional difficulty for introducing structural relations between skin and the other features. As an example, while it is possible to define structural relations such as “the pupil is above the nostrils”, it would be more difficult to define a similar relation with respect to the skin.

The face images used in our experiments have been acquired using a standard digital camera, as well as faces from standard public databases available on-line. We used images from the University of Stirling, available at `http://pics.psych.stir.ac.uk/`. This database has been used for testing face analysis and recognition methods in other papers in the literature, presenting similar characteristics than others such as Feret and XM2VTS [38,39]. The tested images show how robust the method is with respect to images presenting different acquisition conditions (i.e. geometry, illumination, distance from the camera, etc.).

[1] An on-line demo can be found at `http://www.vision.ime.usp.br/~cesar/journals/rti04tracking/`

The face region extracted by the GWN undergoes a watershed transform thus producing an over-segmented image (see Fig. 3). Too small regions produced in the over-segmentation are discarded.

4.2. Attributes

The object attributes are calculated from image connected regions while relational attributes are based on the spatial disposition of the regions. The adopted attributes for the experiments presented in this paper are:

- *Object attributes*: Let $G = (N, E, \mu, \nu)$ be an ARG and let $a \in N$. The set of object attributes $\mu(a)$ is defined as $\mu(a) = (g(a), g_{min}(a), l(a))$, where $g(a)$ denotes the average gray-level of the image region associated to vertex a, $g_{min}(a)$ denotes the average gray-level of the 15% darkest pixels of the image region associated to vertex a, and $l(a)$ is a region label assigned with respect to the approximative landmarks provided by the tracking procedure. Both $g(a)$ and $g_{min}(a)$ are normalized between 0 and 1 with respect to the minimum and maximum possible grey-levels, and the value of 15% used to calculate $g_{min}(a)$ is a parameter that may be changed depending on the test images. The attribute $g_{min}(a)$ has been included in order to facilitate the recognition of textured regions composed of light and dark pixels, such as the eyebrows and, in some cases, the mouth. This feature was particularly important in the experiments performed in this paper. The existence of both light and dark pixels in such facial features may lead to higher average values (i.e. $g(a)$), and the addition of g_{min} avoids confusion with brighter regions like the skin. The region label $l(a)$ indicates which approximative landmark (i.e. left eye, right eye, nose, mouth) is the closest to the region centroid. The region label attribute is particularly useful for constraining the search tree, which is implemented by the dissimilarity measure between object attributes.
- *Relational attribute*: Let $a, b \in N$ be any two vertices of G, and p_a and p_b be the centroids of the respective corresponding image regions. The relational attribute $\nu(a, b)$ of $(a, b) \in E$ is defined as the vector $\nu(a, b) = (p_b - p_a)/(2d_{max})$, where d_{max} is the largest distance between any two points of the considered image region. Clearly, $\nu(a, b) = -\nu(b, a)$.

4.3. Dissimilarity measures

There are two dissimilarity measures c_N and c_E used by the objective function f_1 (Eq. (1)), associated respectively to vertices and edges of the association graph . The measure c_N is related to object attributes, while c_E is related to relational attributes. Their definitions (chosen for this application) are given below.

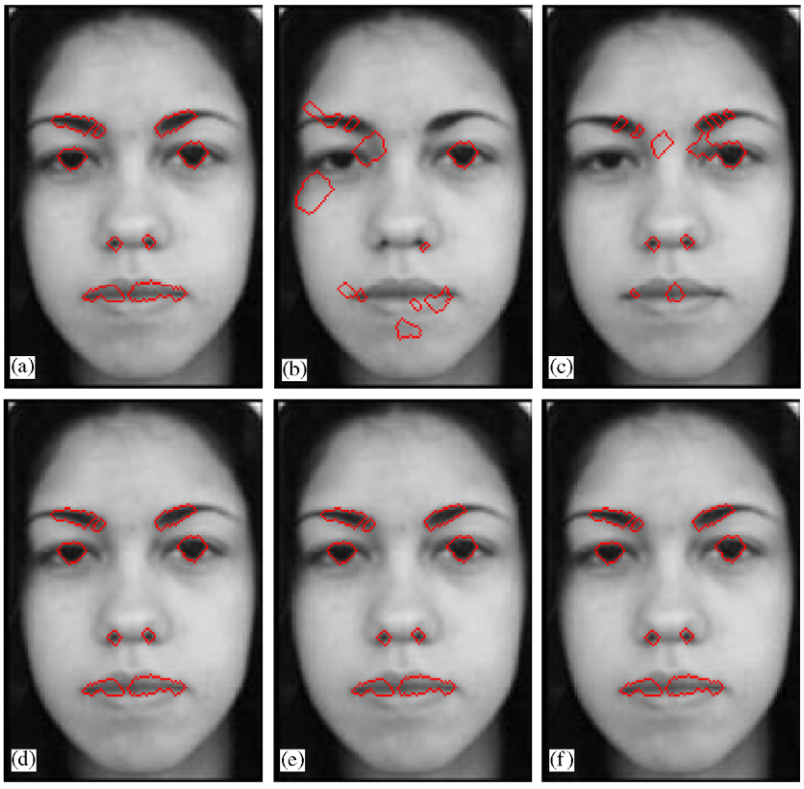

Fig. 4. Segmentation and recognition of facial features using (a) deterministic tree search, (b) eGA, (c) ssGA, (d) UMDA, (e) MIMIC and (f) EBNA. In this example, the target and the model image are the same.

Let (a_1, a_2) denote a vertex of $\tilde{G}_A$, with $a_1 \in N_1$ and $a_2 \in N_2$. The dissimilarity measure $c_N(a_1, a_2)$ is defined as

$$c_N(a_1, a_2) = \times \begin{cases} \gamma_N |g_1(a_1) - g_2(a_2)| & \\ \quad +(1-\gamma_N)|g_{min\,1}(a_1) & \\ \quad -g_{min\,2}(a_2)|, & \text{if } l_1(a_1) = l_2(a_2), \\ \infty & \text{otherwise,} \end{cases}$$

where $(g_i(a_i), g_{min\,i}(a_i), l_i(a_i))$ are the object attributes of G_i, $i = 1, 2$. The parameter γ_N $(0 \leqslant \gamma_N \leqslant 1)$ controls the weights of g and g_{min}. It is worth noting that, if the vertices a_1 and a_2 correspond to regions associated to different approximative landmarks, then the dissimilarity measure equals ∞, and this means that f_1 is not evaluated in such cases. This fact is important because it allows the optimization algorithm to avoid exploring non-desirable solutions such as trying to classify a region near the left eye approximative landmark as mouth, for instance.

Let e denote an edge of $\tilde{G}_A$, with end-points $(a_1, a_2) \in N_A$, $a_1 \in N_1$ and $a_2 \in N_2$ and $(b_1, b_2) \in N_A$, $b_1 \in N_1$ and $b_2 \in N_2$. We compute the modulus and angular differences between $v(a_1, b_1)$ and $v(a_2, b_2)$ as

$$\phi_m(e) = |\|v(a_1, b_1)\| - \|v(a_2, b_2)\||$$

and

$$\phi_a(e) = \frac{|\cos(\theta) - 1|}{2},$$

where θ is the angle between $v(a_1, b_1)$ and $v(a_2, b_2)$, i.e. $\cos(\theta)$ is calculated as

$$\cos(\theta) = \frac{v(a_1, b_1) \cdot v(a_2, b_2)}{\|v(a_1, b_1)\| \|v(a_2, b_2)\|}.$$

The dissimilarity measure $c_E(e)$ is defined as

$$c_E(e) = \gamma_E \phi_a(e) + (1 - \gamma_E)\phi_m(e),$$

where $v(a_i, b_i)$ is the relational attribute (i.e. vector) associated to edge $(a_i, b_i) \in E_i$. The parameter γ_E $(0 \leqslant \gamma_E \leqslant 1)$ controls the weights of ϕ_m and ϕ_a. It is important to note that $v(a, a) = \vec{0}$. This fact means that, when two vertices in G_1 are mapped onto a single vertex of G_2 by the homomorphism, we have $c_E(e) = \|v(a_1, b_1) - \vec{0}\| = \|v(a_1, b_1)\|$, which is proportional to the distance between the centroids of the corresponding regions in the over-segmented image (in such cases, we define $\cos(\theta) = 1$). Therefore, c_E would give large dissimilarity measures when assigning the same label (i.e. the target vertex in G_2) to distant regions and lower

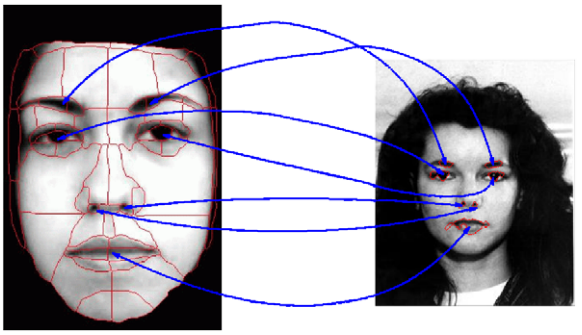

Fig. 5. Correspondences between model and target image recognized facial features, used to calculate the affine transform to match the model mask over the target image.

measures when assigning the same label to near regions, which is intuitively desirable in the present application.

Note that when using f_1 as function to be minimized, c_N and c_E have to be evaluated only for pairs of vertices and edges actually existing in $\tilde{G}_S$.

4.4. Algorithm evaluation

An ARG has been obtained for each input image and a homomorphism has been found using the search algorithms described in Section 3. The obtained results are shown in Fig. 4, showing the obtained segmentation and recognition of the eyebrows, nostrils and lips using the different search algorithms. As it can be seen, the method is able to correctly recognize the facial features of interest, and is robust to substantial differences between the model (Fig. 2(d)) and the target image. A problem that we have experienced here is that the outer portions of the eyebrows in the model contain several skin pixels, leading to misclassifications near it. Therefore, we have identified as "eyebrows" only the two inner portions that compose each eyebrow in the model. Because of the structural constraints in the objective function, the outer portions of the eyebrows in the obtained results have not been included, which is a drawback that we intend to circumvent in future work.

It is interesting to note that the recognized facial features can be used in order to match the model over the input face, which is suitable for visualization purposes and visual assessment of the matching process. Firstly, the centroids of some a priori defined regions of the model are calculated. In the present case, we have chosen to use the eyebrows, pupils, nostrils and lips. The centroids of the respective regions are also calculated for the target image. It is worth noting that these regions have been recognized by the homomorphism, as previously explained. Then, the affine transformation that better maps the model centroids onto the corresponding target image centroids is calculated [40]. Fig. 5 illustrates the correspondences between model and

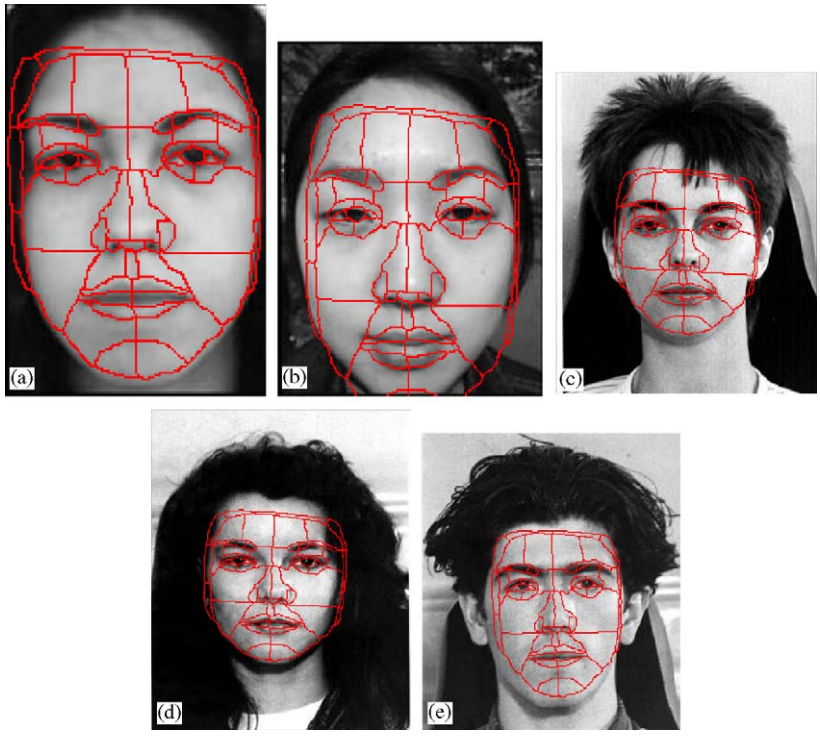

Fig. 6. Model mask matched over the target images using the recognized facial features.

Table 1
Figures of the 6 cases that we analyzed, illustrating the mean values after 5 executions of each of the algorithms (except from the tree search algorithm, which is deterministic)

	bebie1			dani		
	Best	Time	Eval.	Best	Time	Eval.
TreeS	0.326	00:05	—	0.33	00:05	—
eGA	0.457	00:54	202,000	0.463	00:52	202,000
ssGA	0.381	00:40	202,000	0.384	00:37	202,000
UMDA	0.360	00:31	184,710	0.326	00:28	173,914
MIMIC	0.323	00:38	163,119	0.326	00:35	162,720
EBNA	0.323	02:49	175,913	0.326	02:38	163,119
	deise			f014		
	Best	Time	Eval.	Best	Time	Eval.
TreeS	0.313	00:01	—	0.318	00:05:21	—
eGA	0.434	00:33	202,000	0.464	01:30:35	202,000
ssGA	0.357	00:26	202,000	0.388	01:07:50	202,000
UMDA	0.310	00:14	143,909	0.317	00:58:26	184,894
MIMIC	0.311	00:17	129,008	0.317	01:07:46	167,117
EBNA	0.310	01:19	153,924	0.317	04:40:16	185,908
	f041			m036		
	Best	Time	Eval.	Best	Time	Eval.
TreeS	0.322	00:11	—	0.323	00:23	—
eGA	0.463	01:35	202,000	—	—	—
ssGA	0.389	01:09	202,000	0.402	01:54	202,000
UMDA	0.319	00:58	196,702	0.321	01:41	201,900
MIMIC	0.319	01:10	181,910	0.321	02:08	201,900
EBNA	0.319	05:15	195,103	0.321	09:11	201,900

The *best* column corresponds to the mean best fitness value obtained through the search. The *time* column shows the CPU time required for the search (in hh:mm format), and the *eval.* one shows the number of individuals that had to be evaluated in order to end the search.

target image centroids. This affine transformation is then applied to the model mask, which is thus projected onto the target image. Some obtained results are shown in Fig. 6.

The results obtained from the different executions of the algorithms are discussed below. Six different face images were analyzed, and Table 3 shows for each of them the number of regions (nodes) and edges after the automatic over-segmentation procedure. The model used is shown in Fig. 2 and it contains 62 nodes and 3844 edges.

We executed 5 times each of the stochastic algorithms for each of the examples, and in Table 1 the results are given in the form of the mean fitness value of the best individual at the last generation, the CPU time, and the number of different individuals created during the search. The latter computation time is presented as a measure to illustrate the difference in computation complexity of all the algorithms. The machine in which all the executions where performed is a two processor Ultra 80 Sun computer under Solaris version 7 with 1 Gb of RAM. Fig. 7 illustrates the mean performance of the search process of GAs and EDAs during the different generations.

The null hypothesis of the same distribution densities was tested (non-parametric tests of Kruskal–Wallis and Mann–Whitney) for each of the examples and algorithm executions with the statistical package S.P.S.S. release 10.1. The results of these tests are shown in Table 2, and they confirm the significance of the differences of all the algorithms regarding the value of the best solution obtained of EDAs and GAs. They also show that differences between the different EDAs in the best individual obtained are not statistically significant, but these are significant among eGA and ssGA. In all the examples the EDAs obtained better results than the GAs, and these differences are statistically significant regarding Table 2. Also, the differences in execution time are also significant in all the algorithms, and EDAs required always more time. As a result, regarding Tables 1 and 2, we can conclude that the results are much better in EDAs at the expense of a higher computation time, but EDAs arrive to a more satisfactory final individual by having to evaluate less individuals than GAs. This fact is important to take into account if the computation of the fitness function is more complex (i.e. if it

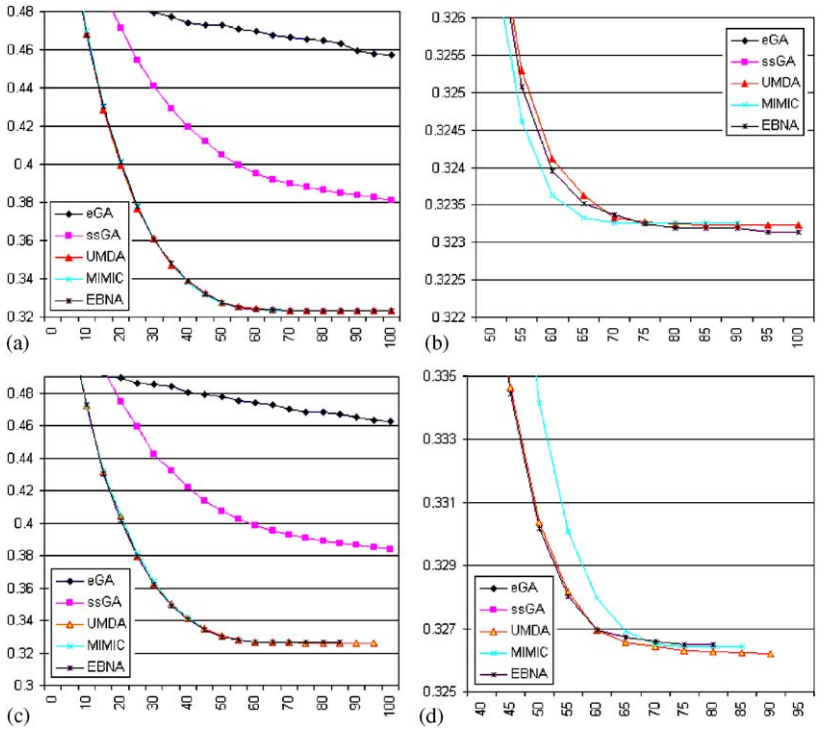

Fig. 7. Comparison on the performance of GAs and EDAs for the examples of *bebie1*, (a) and (b), and *dani*, (c) and (d), showing the fitness of the best individual along the search process. The *x* axis corresponds to the fitness function and the *y* axis to the generation number. The plots (b) and (d) have been added to note more clearly the difference between the EDAs. These graphs show that the performance of all the EDAs is better than such of GAs during the whole research process.

requires more CPU time) than the one selected for our experiments.

In the light of the results obtained for the fitness values, we can conclude the following: generally speaking, EDAs obtained in all the executions a fitter individual than tree search and GAs, but although the number of individuals created is lower than GAs, the CPU time required was bigger.

It is worth emphasizing that one of the main goals of this work is to present a comparative performance assessment of some state-of-the-art optimization methods that may be incorporated in the proposed framework, showing how these methods compare with respect to accuracy of the obtained results and to computation time as well. In this research phase, we have neither aimed at implementing optimized algorithms with respect to computation time nor concentrated efforts on finding out if there are efficient algorithms to implement the proposed framework, which will be done in due time.

Besides the important problem of assessing the optimization algorithms with respect to the objective function and execution time, it is also important to analyze the obtained results with respect to the problem context, i.e. recognition of the facial features of interest. In order to perform this task, we have generated a ground-truth for some faces by manually labelling the over-segmented images, i.e. by obtaining a human solution for the problem. Of course, such ground-truth is prone to some subjectivity, since a human operator decides the label of each region of the over-segmented image with respect to the model (nearby regions around the facial features are generally difficult to be classified, being often labeled differently depending on the operator). Then, the ground-truth is compared to each automatically labeled image to obtain the number of errors, i.e. number of misclassified regions (with respect to the ground-truth). Some of the error regions typically obtained in our experiments are shown in Fig. 8. This figure shows 3 types of errors found in our experiments:

- very small regions nearby the facial features (indicated by “A” in Fig. 8), which have generally been missed by the operator while producing the ground-truth;
- regions in the outer portion of the eyebrows (indicated by “B” in Fig. 8), which have been left out during the classification procedure because of the above explained reasons;
- true errors such as the region indicated by “C” in Fig. 8.

Table 4 shows the obtained errors for some images. We have performed this ground-truth-based assessment (total number of misclassified regions and percentage with respect to the

Table 2
Statistical significance for all the 6 examples and algorithms, by means of the results of the non-parametric tests of Kruskal–Wallis and Mann–Whitney

		GAs-EDAs	Among GAs	Among EDAs
bebie1	Best	$p < 0.001$	$p = 0.008$	$p = 0.620$
	Eval.	$p < 0.001$	$p = 1.000$	$p = 0.084$
	Time	$p = 0.016$	$p = 0.008$	$p < 0.001$
dani	Best	$p < 0.001$	$p = 0.008$	$p = 0.677$
	Eval.	$p < 0.001$	$p = 1.000$	$p = 0.063$
	Time	$p = 0.177$	$p = 0.008$	$p = 0.002$
deise	Best	$p < 0.001$	$p = 0.008$	$p = 0.078$
	Eval.	$p < 0.001$	$p = 1.000$	$p = 0.068$
	Time	$p = 0.121$	$p = 0.008$	$p < 0.001$
f014	Best	$p < 0.001$	$p = 0.008$	$p = 0.105$
	Eval.	$p < 0.001$	$p = 1.000$	$p = 0.064$
	Time	$p = 0.495$	$p = 0.008$	$p = 0.002$
f041	Best	$p < 0.001$	$p = 0.008$	$p = 0.811$
	Eval.	$p < 0.001$	$p = 1.000$	$p = 0.012$
	Time	$p = 0.643$	$p = 0.008$	$p < 0.002$
m036	Best	$p < 0.001$	—	$p = 0.085$
	Eval.	$p < 0.001$	—	$p = 1.000$
	Time	$p = 0.306$	—	$p = 0.002$

The first column shows the result of the test comparing all EDAs with all GAs, the second is the test for comparing eGA and ssGA, and the third is the comparison between the three EDAs.

Table 3
Figures of the 6 cases that we analyzed, illustrating the number of nodes and arcs that are considered

	bebie1	dani	deise	f014	f041	m036
Nodes	147	148	112	176	183	228
Arcs	21609	21904	12544	30976	33489	51984

total number of regions in the segmented image) for the different optimization algorithms.

As it can also be seen, the ssGA and eGA algorithms lead to much poorer results with respect to the recognized facial features and the best solution obtained. This result can be partially understood by the fact that these are both general purpose algorithms. The use of GAs specially thought for our problem could lead to better results. However, it must also be said that the EDAs applied are also general purpose ones.

It should be noted that all these results should be considered only from the optimization point of view and for comparison purpose. It is clear that absolute results could be improved by introducing more attributes, in order to guarantee that the optimum of f_1 actually corresponds to the expected solution (with no error).

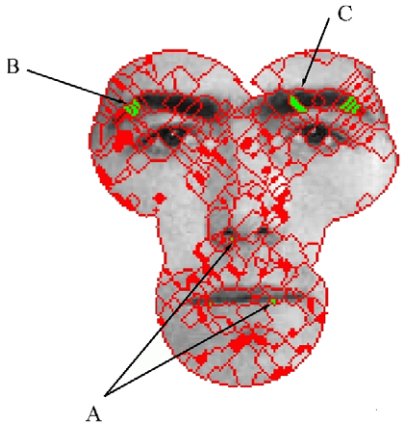

Fig. 8. Example of some typical error regions.

Table 4
Number of misclassified regions in each test image for each algorithm

	deise		f014		f041		m036	
	Errors	%	Errors	%	Errors	%	Errors	%
TreeS	2	1.79	9	5.11	6	3.28	10	4.39
eGA	21	18.75	36	20.45	46	25.14	50	21.93
ssGA	35	31.25	63	35.8	57	31.15	—	—
UMDA	1	0.89	12	6.82	7	3.83	12	5.26
MIMIC	1	0.89	12	6.82	8	4.37	15	6.58
EBNA	1	0.89	11	6.25	5	2.73	15	6.58

The "Errors" column indicates the number of misclassified regions, while "%" shows the percentage with respect to the total number of regions in the image.

5. Concluding remarks and future work

A new method for model-based recognition in images has been proposed, by expressing the problem as an inexact graph matching problem and its optimization through an objective function. New algorithms have been proposed and other ones have been adapted to this problem. As an illustration, we described a new approach for facial feature segmentation based on graph homomorphisms: we have defined ARG representations of a face model and the image to be recognized, an objective function, and applied the optimization algorithms in order to evaluate and compare them.

Our ongoing work aims at improving the method robustness and at generalizing it in a number of different ways,

e.g. using fuzzy morphisms [41] and developing other object and relational attributes. A foreseen extension is to adapt the method to time varying images such as video sequences, by taking advantage of the homomorphism found in a frame in a video sequence when searching for the one in the next frame. Such a strategy could explore a model parameter update procedure, and of extended graphs including "temporal" edges linking regions in successive images.

Furthermore, the above definition of graph homomorphism implies that all vertices in G_1 are mapped to G_2 and if the input image presents features not known by the model, they will be classified. For instance, in the face application, if the input face has glasses and the model graph does not include them, the glass regions will be classified as skin or some other facial feature. Two possible solutions to this problem can be designed. The first one is to leave some of the G_1 vertices unmapped. The second approach is to define an "unknown label vertex" in the model graph, to which the unclassified input regions should be mapped. Both approaches present specific difficulties and are currently being considered in our research. Also other objective functions are currently under investigation.

Acknowledgements

R. Cesar and I. Bloch are grateful to CAPES-COFECUB. R. Cesar is grateful to FAPESP (99/12765-2) and to CNPq (300722/98-2). E. Bengoetxea, I. Bloch and P. Larrañaga would also like to thank The University of the Basque Country (grant 9/UPV 00140.226-15334/2003) and the French Ministry for Education, Research and Technology (grant PICASSO-00773TE). The authors are grateful to the University of Bern and to the University of Stirling for maintaining the face image databases used in this paper. Finally, the authors are grateful to the reviewers who helped improving the final version of the paper with useful comments.

References

[1] B.T. Messmer, H. Bunke, Efficient subgraph isomorphism detection: a decomposition approach, IEEE Trans. Knowledge Data Eng. 12 (2) (2000) 307–323.

[2] R.C. Wilson, E.R. Hancock, A Bayesian compatibility model for graph matching, Pattern Recognition Lett. 17 (3) (1996) 263–276.

[3] A.D.J. Cross, E.R. Hancock, Convergence of a Hill Climbing Genetic Algorithm for Graph Matching, Lecture Notes in Computer Science, vol. 1654, 1999.

[4] A.W. Finch, R.C. Wilson, E.R. Hancock, Matching Delaunay graphs, Pattern Recognition 30 (1) (1997) 123–140.

[5] S. Gold, A. Rangarajan, A graduated assignment algorithm for graph matching, IEEE Trans. Pattern Anal. Mach. Intell. 18 (4) (1996) 377–388.

[6] A.D.J. Cross, E.R. Hancock, Graph matching with a dual-step EM algorithm, IEEE Trans. Pattern Anal. Mach. Intell. 20 (11) (1998) 1236–1253.

[7] A.W. Finch, R.C. Wilson, E.R. Hancock, Symbolic graph matching with the EM algorithm, Pattern Recognition 31 (11) (1998) 1777–1790.

[8] P. Larrañaga, J.A. Lozano (Eds.), Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation, Kluwer Academic Publishers, Dordrecht, 2001.

[9] A. Perchant, C. Boeres, I. Bloch, M. Roux, C. Ribeiro, Model-based scene recognition using graph fuzzy homomorphism solved by genetic algorithm, in: GbR'99 Second International Workshop on Graph-Based Representations in Pattern Recognition, Castle of Haindorf, Austria, 1999, pp. 61–70.

[10] A. Deruyver, Y. Hodé, Constraint satisfaction problem with bilevel constraint: application to interpretation of over-segmented images, Artif. Intell. 93 (1997) 321–335.

[11] M.L. Williams, R.C. Wilson, E.R. Hancock, Deterministic search for relational graph matching, Pattern Recognition 32 (1999) 1255–1271.

[12] J.-P. Braquelaire, L. Brun, Image segmentation with topological maps and inter-pixel representation, J. Visual Commun. Image Representat. 9 (1) (1998) 62–79.

[13] H. Bunke, Error correcting graph matching: on the influence of the underlying cost function, IEEE Trans. Pattern Anal. Mach. Intell. 21 (9) (1999) 917–922.

[14] M. Skomorowski, Use of random graph parsing for scene labelling by probabilistic relaxation, Pattern Recognition Lett. 20 (1999) 949–956.

[15] E. Bengoetxea, P. Larrañaga, I. Bloch, A. Perchant, C. Boeres, Inexact graph matching by means of estimation of distribution algorithms, Pattern Recognition 35 (2002) 2867–2880.

[16] M. Pantic, L.J.M. Rothkrantz, Automatic analysis of facial expressions: the state of the art, IEEE Trans. Pattern Anal. Mach. Intell. 22 (12) (2000) 1424–1445.

[17] R. Brunelli, T. Poggio, Face recognition: features versus templates, IEEE Trans. Pattern Anal. Mach. Intell. 15 (10) (1993) 1042–1052.

[18] R. Herpers, G. Sommer, An attentive processing strategy for the analysis of facial features, in: H. Wechsler et al. (Ed.), Face Recognition: From Theory to Applications, NATO ASI Series F, vol. 163, Springer, Berlin, 1998, pp. 457–468.

[19] E. Bengoetxea, P. Larrañaga, I. Bloch, A. Perchant, Solving graph matching with EDAs using a permutation-based representation, in: P. Larrañaga, J.A. Lozano (Eds.), Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation, Kluwer Academic Publisher, Boston, Dordrecht, London, 2001, pp. 239–261, (Chapter 12).

[20] A. Perchant, I. Bloch, A new definition for fuzzy attributed graph homomorphism with application to structural shape recognition in brain imaging, in: IMTC'99 16th IEEE Instrumentation and Measurement Technology Conference, Venice, Italy, 1999, pp. 1801–1806.

[21] R.M. Cesar Jr., I. Bloch, First results on facial feature segmentation and recognition using graph homomorphisms, in: VI Simpósio Ibero-Americano de Reconhecimento de Padr oes SIARP 2001, Florianapolis, Brazil, 2001, pp. 95–99.

[22] R.M. Cesar Jr., E. Bengoetxea, I. Bloch, Inexact graph matching using stochastic optimization techniques for facial, in: Proceedings of the International Conference on Pattern Recognition, ICPR 2002, vol. 2, Québec, Canada, 2002, pp. 465–468.

[23] J.H. Holland, Adaptation in Natural and Artificial Systems, The University of Michigan Press, Michigan, 1979.

[24] D.E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison/Wesley, Reading, MA, 1989.
[25] Z. Michalewicz, Genetic Algorithms+Data Structures = Evolution Programs, Springer, Berlin, Heidelberg, 1992.
[26] D. Whitley, J. Kauth, GENITOR: a different genetic algorithm, in: Proceedings of the Rocky Mountain Conference on Artificial Intelligence, vol. 2, 1988, pp. 118–130.
[27] H. Mühlenbein, T. Mahning, The factorized distribution algorithm for additively decomposed functions, in: Second Symposium on Artificial Intelligence Adaptive Systems, CIMAF 99, Havana, Cuba, 1999, pp. 301–313.
[28] S. Baluja, Population-based incremental learning: a method for integrating genetic search based function optimization and competitive learning, Technical Report, Carnegie Mellon Report, CMU-CS-94-163, 1994.
[29] H. Mühlenbein, G. Paaß, From recombination of genes to the estimation of distributions i. Binary parameters, in: M. Voigt, W. Ebeling, I. Rechenberg, H.-P. Schwefel (Eds.), Parallel Problem Solving from Nature—PPSN IV, Lecture Notes in Computer Science, vol. 1411, 1996, pp. 178–187.
[30] E. Bengoetxea, P. Larrañaga, I. Bloch, A. Perchant, C. Boeres, Learning and simulation of Bayesian networks applied to inexact graph matching, Pattern Recognition 35 (12) (2002) 2867–2880.
[31] M. Henrion, Propagating uncertainty in Bayesian networks by probabilistic logic sampling, in: J.F. Lemmer, L.N. Kanal (Eds.), Uncertainty in Artificial Intelligence, vol. 2, North-Holland, Amsterdam, 1988, pp. 149–163.
[32] H. Mühlenbein, The equation for response to selection and its use for prediction, Evolut. Comput. 5 (1998) 303–346.
[33] S. Baluja, S. Davies, Using optimal dependency-trees for combinatorial optimization: Learning the structure of the search space, Technical Report, Carnegie Mellon Report, CMU-CS-97-107, 1997.
[34] J.S. de Bonet, C.L. Isbell, P. Viola, MIMIC: finding optima by estimating probability densities, in: M.C. Mozer, M.I. Jordan, T. Petsche (Eds.), Advances in Neural Information Processing Systems, vol. 9, The MIT Press, Cambridge, MA, 1997, pp. 424–431.
[35] R. Etxeberria, P. Larrañaga, Global optimization with Bayesian networks, in: Special Session on Distributions and Evolutionary Optimization, II Symposium on Artificial Intelligence, CIMAF99, 1999, pp. 332–339.
[36] V. Kruger, G. Sommer, Affine real-time face tracking using a wavelet network, in: Proceedings of ICCV'99 Workshop Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, Corfu, Greece, 1999, pp. 141–148.
[37] R.S. Feris, V. Krueger, R.M. Cesar Jr., Efficient real-time face tracking in wavelet subspace, in: Proceedings of the Second International Workshop on Recognition, Analysis and Tracking of Faces and Gestures in Real-time Systems, Eighth IEEE International Conference on Computer Vision, RATFG-RTS—ICCV 2001—Vancouver, Canada, 2001, pp. 113–118.
[38] P.J. Phillips, H. Moon, P.J. Rauss, S. Rizvi, The FERET evaluation methodology for face recognition algorithms, IEEE Trans. Pattern Anal. Mach. Intell. 22 (10) (2000) 1090–1104.
[39] XM2VTSDB. The xm2vts database, http://www.ee.surrey.ac.uk/Research/VSSP/xm2vtsdb/
[40] L.F. Costa, R.M. Cesar Jr., Shape Analysis and Classification: Theory and Practice, CRC Press, Boca Raton, FL, 2001.
[41] A. Perchant, I. Bloch, Fuzzy morphisms between graphs, Fuzzy Sets and Systems 128 (2) (2002) 149–168.

About the Author—ROBERTO M. CESAR Jr. received a B.Sc. in Computer Science ("Universidade Estadual Paulista"—"UNESP"), a M.Sc. in Electrical Engineering ("Universidade de Campinas"—"UNICAMP") and a Ph.D. in Computational Physics at the Institute of Physics, USP—"Universidade de São Paulo", Brazil. His main research topics are in shape analysis, computer vision and structural pattern recognition. He is currently an associated professor at the Department of Computer Science of IME—USP.

About the Author—ENDIKA BENGOETXEA is Lecturer at the Computer Architecture and Technology Department at the University of the Basque Country. He obtained his B.Sc. in Computer Science from the University of the Basque Country and Brighton University in 1994, and his M.Sc. in Medical Imaging from the University of Aberdeen in 1999, and his Ph.D. from the Ecole Nationale Supérieure des Télécommunications (ENST) in Paris. His research interests reside in evolutionary algorithms, graph matching, as well as in applying pattern recognition and classification techniques to medical data and images.

About the Author—ISABELLE BLOCH is Professor at ENST (Signal and Image Processing Department). Her research interests include 3D image and object processing, 3D and fuzzy mathematical morphology, decision theory, information fusion, fuzzy set theory, belief function theory, structural pattern recognition, spatial reasoning, medical imaging.

About the Author—PEDRO LARRAÑAGA is Professor in the Departement of Computer Science and Artificial Intelligence at the University of the Basque Country where he has been leading the Intelligent Systems Group since 1996. He is author or coauthor of three books and more than 70 chapters and papers in refereed journals in the fields of probabilistic graphical models and evolutionary computation with application in medicine and computational biology. Recently he has served as guest editor for the journals Artificial Intelligence in Medicine, Evolutionary Computation and Machine Learning.