TABLE 1
Comparison between Classical Graph-Matching Methods in Terms of Their Computational Complexity and the Ability to Perform an Inexact Matching

	Graph Isomorphism	Subgraph Isomorphism	Error-tolerant Subgraph Isomorphism	Optimal	Complexity Class	Key References
Backtrack tree search	Yes	Yes	No	Yes	NP	
Forward checking	Yes	Yes	No	Yes	NP	[32]
Discrete relaxation	Yes	Yes	Yes[1]	Yes	NP[2]	[12]
Association graphs	Yes	Yes	No	Yes	NP	[14, 23]
Graph edition	Yes	Yes	Yes	Yes	NP	[7, 21, 36]
Random graphs	Yes	Yes	Yes	Yes	NP	[25, 38]
Probabilistic relaxation	Yes	Yes	Yes	No	P	[5, 8, 11, 37]
Neural networks	Yes	Yes	Yes	No	P	[16, 29, 28]
Genetic algorithms	Yes	Yes	Yes	No	P	[6, 9, 15]
Eigendecomposition	Yes	No	No[3]	Yes	P	[33]
Linear programming	Yes	No	No	Yes	P	[2]
Indexed search	Yes	Yes	No	Yes	P[4]	[4, 27]

[1] In some cases (e.g. [12]).
[2] If backtracking follows relaxation.
[3] Although is able to find error-tolerant graph isomorphism between close graphs.
[4] Although the compilation of the database is NP.

relaxation procedure. Another continuous optimization approach is based on *neural networks* [16], [28], [29]. The nodes of a neural network can represent vertex-to-vertex mappings and the connection weights between two network nodes represent a measure of the compatibility between the corresponding mappings. The network is programmed in order to minimize an energy (cost) function which is defined in terms of the compatibility between mappings. The problem of neural networks is that the minimization procedure is strongly dependent on the initialization of the network. *Genetic algorithms* is another technique used to find the best match between two graphs [6], [9], [15]. Vectors of *genes* are defined to represent mappings from model vertices to input vertices. These solution vectors are combined by genetic operators to find a solution.

Algebraic Algorithms. *Weighted graphs* are a particular type of graphs which have weights assigned to their edges. A weighted graph G can be represented by an adjacency matrix $A(G)$, where a position a_{ij} contains the weight associated with the edge (v_i, v_j). An analytic approach can then be used to solve the matching problem. Interesting approaches based on algebraic manipulations of adjacency matrices were respectively proposed by Umeyama [33] (eigendecomposition) and Almohamad and Duffuaa [2] (linear programming). These methods only work when both model and input graphs have the same number of nodes.

Indexed Search. Graph matching is also used in content-based image retrieval. When image concepts can be represented by graph structures, the problem of looking in the database for an image that contains a certain object can be stated in terms of an indexed search in a graph database. Sossa and Horaud [27] proposed using the second immanantal polynomial of the Laplacian matrix of a graph for hash-coding any graph. Bunke and Messmer [4] proposed a decision tree approach organized in terms of the different permutations of the adjacency matrices of the graphs in the database. Indexed search usually requires an exponential complexity for the compilation of the database.

In this paper, an error-tolerant subgraph isomorphism algorithm for symbol recognition in a graphics recognition framework is described. Graphics recognition is a growing area within the document analysis field devoted to recognizing graphical entities in printed documents such as maps, plans, diagrams, engineering drawings, etc. Graph matching methods are often proposed to solve the problem of recognizing symbols in line drawings or diagrams [12], [16], [17]. The problem consists in finding a model graph that represents the prototype symbol as a subgraph of an input graph representing the diagram. We propose a two-level graph representation. In the first level, the vectorized document is approximated by graphs whose nodes represent characteristic points (junctions, end, or corner points) and whose edges approximate the segments between characteristic points. In the second level, data is organized in terms of *Region Adjacency Graphs* (RAG). The RAG nodes represent the regions, i.e., minimal closed loops of the first-level graphs, and the edges are the neighboring relations between loops. A given first-level graph[1] region (RAG node) is encoded by its shape description using an attributed cyclic string containing the sequence of graph edges defining the region. Symbols are then recognized in terms of an inexact graph-matching procedure that computes the minimum distance from a model RAG to an input RAG. This distance is considered to be the weighted sum of the costs of edit operations to transform one graph into the other. The distance between RAGs conveys also a distance computation between regions. Since region boundaries are encoded by strings, their similarity is defined in terms of a string edit distance formulation [34].

Since the matching algorithm is formulated in terms of an error-tolerant subgraph isomorphism, it can also be applied to disturbed documents such as hand drawn, inaccurately vectorized, or documents scanned with an insufficient resolution. In addition, our graph-matching algorithm, proposed in the field of graphics recognition, could be extended to other object recognition frameworks whenever a previous segmentation stage is assumed, from which regions are extracted from the input image according to a homogeneity criterion such as color or texture, a Voronoi tessellation of the image plane, etc. Thus, although the low-level procedure to extract regions from the input image is application-dependent, the graph-matching algorithm proposed in this paper is application-independent.

1. In the remainder of this paper, first-level attributed graphs will be referred to as *plain graphs* or just *graphs* to avoid confusion with RAGs.