ABSTRACT
Graph representation and graph matching have been successfully applied to a large number of problems in computer vision and pattern recognition. Concerning graph matching, the classical algorithms of graph isomorphism seems useless when the image is degraded with noise or vectorial distortion. This paper introduce a novel similarity measure to recognize symbols by performing inexact matching of attributed graphs. In the proposed approach, symbols are encoded by attributed graphs, whose nodes represent structural primitives like quadrilaterals and whose edges represent mutual relationships between these primitives. To be invariant of rotation and scaling, relative information about primitives are associated as attributes on the nodes and edges. Considering a mapping between two graphs, a similarity function is formulated, that use the numerical values of the attributes to calculate a similarity score. This new similarity measure has many desirable properties such as discrimination power, invariant to affine transformations, and robustness to noise or vectorial distortions.

KEY WORDS
Document image analysis, graphic symbols recognition, attributed relational graph matching.

1. Introduction

Graphics Recognition is a branch of document analysis that focuses on the recovery of graphical information in documents. In the field of document analysis, the word symbol has often been used for alphanumeric characters as well. But the problem of symbol recognition is very different from basic character recognition because of much higher number and variety of symbols to be recognized. Symbols are set of signs or shapes with a particular meaning that often represent something in a specific application domain. The linear graphic symbols (figure. 1) are made of lines, arcs, and simple geometric primitives, which can be found in any kind of graphic drawings. The early specific work on symbol recognition, as opposed to character recognition, emphasized the use of structural pattern recognition techniques, as usual statistical classification techniques were not suitable [1].

A lot of symbol recognition methods can be found in the literature [2], [3], [4], [5], [6] and a review of the current state of the art is presented in [7]. Most of the presented techniques have been applied successfully to a specific domain but very little work has been done on generic domain independent, symbol recognition. This paper presents a new generic symbol recognition system that uses a mix structural and numerical representation of graphics symbols and the newly proposed similarity measure to do an inexact matching of attributed graphs for the symbol recognition purpose. The rest of this paper is organized as follows: Section 2 gives an overview of the symbols representation steps, in Section 3, our graph matching algorithm is discussed. Then in Section 4, we present results and finally, in Section 5, we draw conclusion from this work.

2. A Mix Structural and Numerical Representation of Linear Graphic Symbols

Considering structural methods, in most of the cases, we observe symbolic representations of the shapes to recognize. For example, a method based on vectorial signature for discriminating symbols in technical documents is presented in [5]. The vectorial signatures are built from the occurrences of constraints between segments, as parallelisms, straight angles, overlap ratios, etc. The study of spatial relationship between pair of lines and simply counting number of L-junction, V-junction,
parallel and collinear segments may work well for few symbols of a particular domain but as the number of the symbols to test increases, a lot of false alarm can be observed. These kind of symbolic labelled based signatures have week discrimination power and can only be used for clustering similar symbols together but not for a generic symbol recognition purpose.

2.1. Raster-to-Vector Conversion

Two major vectorization approaches can be stated : those that are based on skeleton or medial axis computation, and others that do not thin the image but match opposite contours of lines. A comparison of these two approaches have been recently presented by Tombre and Tabonne in [8]. The main disadvantage of vectorization methods based on medial axis is that they tend to introduce distortions at line end points and junctions.

Therefore contours are commonly used in order to obtain invariant descriptions of patterns. The contour-matching approaches like [9], [10] were found to be more accurate than skeleton based methods. The technique which we had proposed in [11] for segmenting the graphical images into a set of quadrilaterals works very well in case of linear graphical symbols (figure 2).

For vectorization of contours, we have used a method suggested by Wall [12]. The polygonal approximation of contours is followed by matching of the corresponding vectors in term of slope, distance and area criteria to build the quadrilaterals, representing lines composed by the graphic symbol.

![Figure 2. Linear graphics symbols and their representation by quadrilaterals](image)

2.2. Zone of influence of quadrilateral

Each quadrilateral has attributes like length (ℓ) of the median axis, angles of the two vectors (ν₁, ν₂), width on each side (W₁, W₂) and a zone of influence – shown by dotted rectangle in the following figure.

![Figure 3. Zone of influence of quadrilateral](image)

The zone of influence play a crucial role in graph extraction phase, where the relative information about the quadrilaterals will be used as vertex attribute and spatial relational information between the quadrilaterals lying in this zone of influence will be considered as attributes on edges of the graph. The dimensions of the zone of influence depends on the length (ℓ) and on the two widths of the quadrilateral i.e., Uₓ = ℓ / 4 and Uᵧ = ((W₁ + W₂) / 2) x 4 as shown in figure 3. The computation and analysis of the zone of influence of each quadrilateral extracted from the initial image allow to obtain a graph, representing all the content of a symbol. This phenomenon is further explain in the following section.

2.3. Attributed Relational Graphs Generation

A graph G = (V, E) in its basic form is composed of vertices and edges. Where V is the set of vertices (also called nodes) and E is the set of edges (also known as arcs) of graph G. If two vertices in G, say u, v ∈ V, are connected by an edge e ∈ E, this is denoted by e = (u, v) and two vertices are said to be adjacent or neighbours. Edges are said to be undirected when have no direction, and a graph G containing only such type of edges is called undirected graph. Also, vertices and edges can contain some other information. These are called vertex and edge attributes, and the graph is called attributed relational graph. To encode each graphic symbol as an undirected attributed relational graph, we defined our attributed relational graph G as a 4-tuple : G = (V, E, α, β), where

- V is the finite set of vertices,
- E ⊆ V × V is the set of edges,
- α : V → Aᵥᵢ function assigning attributes to vertices,
- β : E → Aₑᵢ function assigning attributes to edges.

Here, Aᵥᵢ and Aₑᵢ denotes sets of vertex and edge attributes respectively, i is varying from 1 to δ and j is varying from 1 to Ω. While δ and Ω represent the number of attributes associated to a vertex and an edge of the graph respectively.

In the proposed attributed graph, each vertex represents a quadrilateral and stores its relative length as an attribute while each edge is associated with relational information i.e., the angle between the two quadrilaterals.

The relative length λᵢ of the ℓᵢ th quadrilateral of a symbol can be computed as a ratio between its length (ℓᵢ) and the length of the longest quadrilateral (ℓ_max) found in that particular symbol, i.e.,

$$\lambdaᵢ = ℓᵢ / ℓ_max$$

The use of relative length instead of length in pixels allows to be invariant to scaling. This relative length will be stored as a vertex attribute. The angle (ϕᵢ ∈ [0, π]) between the two quadrilaterals (Qᵢ, Qᵢ) can be computed from the angles of their medial axis i.e., if θᵢ, θⱼ are the
angles of $Q_i$ and $Q_j$ with horizontal axis, the angle ($\varphi_{ij}$) is computed as: $\varphi_{ij} = |\theta_i - \theta_j|$.

Figure 4. Angle $\varphi_{ij}$ between quadrilaterals $Q_i$, $Q_j$.

The angle is stored as an edge attribute and its use allows the graph of the symbol to be invariant to rotation. Figure 5, shows how the spatially close quadrilaterals were grouped together in terms of Zone of influence to generate an attributed graph of a symbol. In the figure, only a subgraph generated by the quadrilateral-2 having zone of influence $Z_2$ is shown.

Figure 5. Attributed sub-graph with respect to zone of influence $Z_2$.

3. From Symbolic Graph Matching to Numerical Graphs Matching

3.1. Typical Matching Algorithms

Graph representation and graph matching have been successfully applied to a large number of problems in computer vision and pattern recognition [13]. Graph matching refers to the problem of finding a mapping from the vertices of one graph to the vertices of another graph that satisfy some constraints or optimality criteria. The matching methods can be divided into two broad categories: the first contains exact matching methods that are characterized by the fact that the mapping between the nodes of the two graphs must be edge preserving in the sense that if two nodes in the first graph are linked by an edge, they have to be mapped to two nodes in the second graph that are linked by an edge as well. Graph isomorphism comes under this category if the condition is hold in both direction and the mapping is bijective [14]. A weaker form of exact matching is also subgraph isomorphism.

Most of the algorithms for exact graph matching are based on some form of tree search with backtracking [15], [16]. However, exact graph matching has exponential time complexity in the worst case.

The second category defines inexact matching methods, where a matching can occur even if the two graphs being compared are structurally different to some extent. In many applications, the observed graphs are subject to deformations due to several causes: intrinsic variability of the patterns, noise in the acquisition process etc. Graph matching procedures that allow for such structural variations are so called inexact matching algorithms [17], [18]. In practice with images, inexact matching algorithms are the only effective ones because of noise and shape variability. For example, an error-tolerant sub-graph isomorphism algorithm formulated in terms of Region Adjacency Graphs (RAG) was proposed in [6] to recognize symbols in hand drawn diagrams. The medial axis based vectorization of the image is followed by two-level graph representation. The first-level graph is generated from the characteristics points (junction, end, or corner points) and the segments joining these points. In the second level, regions, i.e., the minimal closed loops of the first level graphs with their adjacency relations, are encoded as Region Adjacency Graph (RAG) and classical string edit distance is used to compute similarity between regions boundaries strings. However, in the presence of noise extra characteristics points can appear, resulting in surplus vertices in the first level graph and ultimately may produce new unexpected RAG.

The concept of graph probing presented in [19] is an interesting combination of structural and statistical approach. Graph comparison is done by correlating the collected statistics from the vertices and edges of the graphs. The different type of information (they called probes) include verifying the occurrences of a given type of vertices, their content and label specifications, in- and out-degrees of the vertices etc. Although graph probing provides a measure of similarity between two graphs or how similar one graph is to some subgraph of another, but does not provide a mapping from one graph to the other.

3.2. The New Numerical Graph Matching

We propose a novel method to compute a similarity measure between graphs. It can be viewed as two steps process, first, choice of a mapping, and second, computation of a similarity score for that specific mapping between the two graphs.

3.2.1. Similarity Measure Computation

To compare two graphs with numerical attributes on vertices and edges. We propose a new distance based measure which is capable of computing vertex-to-vertex similarity and edge-to-edge similarity present in a particular mapping. We propose to calculate the similarity score of a given mapping(Mp) as:

$$S_{C_{Mp}} = \left[ \sum_{i=1}^{m} (1 - \Delta V_i) + \sum_{j=1}^{n} (1 - \Delta E_j) \right] \left( \sum_{i=1}^{k} \omega_i + \sum_{j=1}^{f} \omega_j \right)$$

Where, $m$ is total number of mapped vertices in a mapping and $n$ is the total number of edges in between them.
The \( \omega_i \) and \( \omega'_{j} \) are the weights associated to the split of the \( i^{th} \) vertex and the \( j^{th} \) edge in the given mapping \( (M_p) \) respectively. The splits are the association of a vertex (or an edge) in one graph to more than one vertices (or edges) in the other graph. The weight of the split depends on the number of attributes associated to the vertices or edges.

\[
\Delta V_i = \sum_{i=1}^{\delta} f_i(A_i^1, A_i^2) \quad \text{and} \quad \Delta E_j = \sum_{i=1}^{\Omega} g_i(A_j^1, A_j^2)
\]

In formula (1), \( \Delta V_i \) correspond to the distance between two mapped vertices, normalized between 0 and 1. The function \( f_i \) compares the values of the \( k^{th} \) attribute of the two mapped vertices and return a similarity value between 0 and 1. As in this work, only one attribute has been associated to the vertices (i.e., the relative length of the quadrilateral), we can simply use

\[
f_i(A_i^1, A_i^2) = |\lambda_i - \lambda_i'|
\]

Similarly, \( \Delta E_j \) is corresponding to the distance between two mapped edges normalised between 0 and 1. The function \( g_i \) is used to compare the two values of the \( k^{th} \) attribute of the two edges. Actually, we have used only one attribute to compare the edges (i.e., the relative angle between quadrilaterals), we define \( g_i \) as :

\[
g_i(A_i^1, A_i^2) = \frac{\phi_i - \phi_i'}{180}
\]

To normalize the similarity measure between 0 and 1, we use :

\[
\text{Sim}(G, G') = \frac{S_{C_{M_P}}}{[\delta \times (C(V') + C(V')) + \Omega \times (C(E') + C(E'))]}
\]

Here \( S_{C_{M_P}} \) is the score of the mapping computed according to new similarity formula given in eq. 1. While \( C \) is a cardinality function that return the number of vertices or edges in a graph. Whereas \( \delta \) and \( \Omega \) represent the number of attributes associated to a vertex and an edge of the graph respectively.

### 3.2.2. Finding of Best Mapping between Graphs

The simple way to proceed is to test all the possible mapping between the two graphs and finally keeping the best one (i.e., the one which produced the maximum similarity score). Since the search space often expands in an exponential way as the size of the graphs increases, the demand for computational resources quickly become excessive. As a result, less complex algorithms like “Branch and Bound” or “Greedy Algorithms” are often implemented to avoid exhaustive searches, but at the risk of finding a sub-optimal solution rather than a truly optimal one. The reduction of time complexity and memory occupation can be obtained by suitable look ahead rules during the matching process. Thus, to find the best mapping between two graphs, we use an algorithm inspired from the greedy algorithm presented in [20]. This is implemented in the SimGraph routine described below.

#### 3.2.3. SimGraph routine

SimGraph is the name of the routine which is programmed for finding best mapping between two graphs \( (G, G') \). It takes two attributed graphs as input and return the best mapping \( (M_{best}) \) of vertices. It starts with an empty mapping and in each iteration one candidate couple \((v, v')\) is allowed to enter into the current mapping \( (M_{C}) \) provided that it has the ability to increase the previous similarity score. In case there are more than one candidate couple of vertices which can equally increase the score function, a second level score i.e., step_ahead_score is calculated between those edges of vertices \( v \) and \( v' \) which are not part of the current mapping \( (M_{C}) \). This condition helps to check the effectiveness of a candidate couple if allowed to enter into current mapping \( (M_{C}) \) and to avoid tie between them. Thus the candidate with maximum step_ahead_score will be selected. This process continue till the score has stopped increasing and there exist no more vertices or edges to be added into current mapping.

### 4. Experimental Results

To test our prototype system, we have done experiments using GREC2003 database [21]. Working with 50 different Model symbols (figure. 1), a set of 1100 examples of different levels of distortion, geometric transformations and common noises were generated. Due to invariant attributes on the vertices and edges, the recognition rate was found to be 100% in case of rotated and scaled images. The table. 1 provide similarity scores obtained using our graph matching algorithm between models symbols and their corresponding locally deformed symbols (i.e., rotated and scaled). For the sake of conciseness, only the similarity scores of few prototypes are shown. The table 2 and 3 provide similarity scores obtained using our graph matching algorithm, between models symbols and their corresponding globally deformed symbols (noise and vectorial distortion respectively). For the sake of conciseness, only the similarity scores of some prototypes are shown. In table 2 and 3, we can see that although the similarity score is not always equal to one, but most of the time the maximum similarity score is on the diagonal of the table showing system ability to recall. A summary of the recognition rates is given in Table.4 that includes all types of variations of the 50 models symbols (figure 1).

386
There are many factors which can influence the performance of a symbol recognition method. One thing was found common in all the methods proposed in GREC2003 contest i.e., scalability issue. The methods worked well in a limited number of symbols (e.g. 5 and 20 Model symbols set) but significantly degrade their performance for a database with 50 models symbols. Our method showed highest discrimination power in case of ideal images, a recognition rate of 100% for 50 model symbols set, much better as compare to the participants methods recognition rates which were 92% and 84%.

In case of affine transformation, the propose method results are higher than majority of the participants, which were either influenced by rotation or by scaling. However, as expected, the more the noisy image, the lower the recognition rates are. The reason is, in this work, we have not done any pre-processing such as noise filtering or removal of small quadrilaterals which had results in extra vertices of the graphs. The number of model symbols in case of vectorial distortion test is lower (i.e., only 15) because the method used in GREC03 contest for producing vectorial distortion works only with symbols having straight lines and not arcs. Our recognition rate for these deformed symbols, if not better, is quite close to the best proposed methods.

### Table 1

<table>
<thead>
<tr>
<th>Model Symbol</th>
<th>Query Symbol</th>
<th>Detected correctly</th>
<th>Missed</th>
<th>Recog. Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rotation 50</td>
<td>150</td>
<td>250</td>
<td>242</td>
<td>8</td>
</tr>
<tr>
<td>Scaling 50</td>
<td>100</td>
<td>100</td>
<td>8</td>
<td>100%</td>
</tr>
<tr>
<td>Noise Level 1</td>
<td>50</td>
<td>250</td>
<td>242</td>
<td>8</td>
</tr>
<tr>
<td>Level 2</td>
<td>50</td>
<td>250</td>
<td>238</td>
<td>12</td>
</tr>
<tr>
<td>Level 3</td>
<td>50</td>
<td>250</td>
<td>230</td>
<td>20</td>
</tr>
</tbody>
</table>

### Table 2

<table>
<thead>
<tr>
<th>Model Symbol</th>
<th>Query Symbol</th>
<th>Detected correctly</th>
<th>Missed</th>
<th>Recog. Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rotation 50</td>
<td>150</td>
<td>250</td>
<td>242</td>
<td>8</td>
</tr>
<tr>
<td>Scaling 50</td>
<td>100</td>
<td>100</td>
<td>8</td>
<td>100%</td>
</tr>
<tr>
<td>Noise Level 1</td>
<td>50</td>
<td>250</td>
<td>242</td>
<td>8</td>
</tr>
<tr>
<td>Level 2</td>
<td>50</td>
<td>250</td>
<td>238</td>
<td>12</td>
</tr>
<tr>
<td>Level 3</td>
<td>50</td>
<td>250</td>
<td>230</td>
<td>20</td>
</tr>
</tbody>
</table>

5. **Conclusion and Future Work**

In this paper we presented a new method based on similarity measure, to recognize linear graphic symbols, represented by attributed graph structures. The use of quadrilaterals preserves the significant parts of the symbol.
symbol. Their relative lengths as attributes on vertices and their relative angles with neighbouring quadrilaterals as attributes on edges make our attributed graphs invariant of rotation and scaling. The results obtained under global transformation applied to symbols are very encouraging.

To deal with the presence of noise and distortion, our approach is based on error tolerant inexact graph matching that uses numerical values of the attributes rather than symbolic values (like L-Junction, V-junction, T-Junction, parallel, ...). Furthermore, the system can easily recognize new symbols by simply selecting its model image.

The algorithm is fairly general and has a broad applicability. The proposed method not only compute a degree of similarity for classification purpose but also provide a vertex to vertex mapping between the two graphs. However, there is still much work need to do for enhancing and enriching this framework. We will also consider adding more information as the vertices and edges attributes to recognize more complex graphic symbols and filled shapes.

References

[1] K. Tombre, S. Tabbone, P. Dosch, Musings on Symbol Recognition, In the Proceedings of 6th APR International Workshop on Graphics Recognition, Hong Kong, 2005, 25-34


