Structural Recognition of Hand Drawn Floor Plans

Josep Lladós, Enric Martí
Unitat de Processament d'Imatges i Intel.ligència Artificial
Departament d'Informàtica - Edifici C - Universitat Autònoma de Barcelona.
08193 Bellaterra (Barcelona)
e-mail: josep@upisun1.uab.es, enric@upisun1.uab.es

Abstract

A system to recognize hand drawn architectural drawings in a CAD environment has been developed. In this paper we focus on its high level interpretation module. To interpret a floor plan, the system must identify several building elements, whose description is stored in a library of patterns, as well as their spatial relationships. We propose a structural approach based on subgraph isomorphism techniques to obtain a high-level interpretation of the document. The vectorized input document and the patterns to be recognized are represented by attributed graphs. Discrete relaxation techniques (*AC4* algorithm) have been applied to develop the matching algorithm. The process has been divided in three steps: node labeling, local consistency and global consistency verification. The hand drawn creation causes disturbed line drawings with several accuracy errors, which must be taken into account. Here we have identified them and the AC4 algorithm has been adapted to manage them.

1 Introduction

CAD systems are a tool of great help that solves efficiently technical document's creation and modification tasks. But, what about the reverse problem, converting paper-based drawings for their integration into a CAD environment? The field of *document analysis* deals with this topic through the use of image processing and pattern recognition techniques applied to scanned images of document pages. In this paper we propose a CAD system input technique from hand drawn floor plans. This alternative input technique shows several advantages: it allows storage and modification of paper-based plans and, thus, the user is offered the possibility of creating new documents in a rapid and easy manner.

The system described below is structured according to the three classical levels of any document analysis and understanding system (figure 1): *lexical level*, *syntactic level* and *semantic level*. The lexical level extracts the basic primitives that construct the line drawing (straight lines, circular arcs, junction points, end points, corner points and inflexion points) and their geometrical and topological properties. The syntactic level establishes structural relations between primitives and provides a symbolic representation of the document. In our case, an *attributed graph* [Cha90] representation has been chosen. The aim of the semantic phase is to understand the document and to obtain a high level representation compatible with a CAD system format. This high level representation, in this work, is a *semantic net* [**?**] whose nodes denote high level graphical objects of the document (doors, tables, windows, etc.), and links denote spatial relationships between them (inside, next to, etc.). This level is aided by domain-dependent knowledge, i.e. some models to be recognized, some constraints to be satisfied, etc.

There are several works on line drawing pattern recognition. Works on logic circuit diagram recognition [FWL89], [ST82], [LKG90], [KU86], [Ble84] deal with a limited set of symbols that the system should recognize and that can be easily separated from other graphical elements. Studies on engineering drawing processing [NL90], [SH89], [PLJ91], [BWS88], [KS93] do not carry out a high level interpretation and are mainly concentrated on an efficient vectorization of the document. In works on parcel interpretation of cadastral city maps [MT90], [BCDB*92], [Mad91] the graphical symbol identification consists in searching polygons with given features (texture, number and configuration of edges, etc.). No works on hand drawn floor plans recognition have been found.

In a floor plan drawing the instances of patterns cannot be easily identified, since they can be joined or embedded in other graphic elements. Besides, considering hand-drawn line drawings supposes accepting a certain degree of inaccuracy in their traces. This

Figure 1: A document analysis and understanding system.

implies using uncertainty parameters (thresholds for angles and distances) to achieve accuracy error correction.

This paper is actually devoted to describe the semantic level. This module performs model-based matching. Models are the graphical instances in the paper-based document to be recognized. Structural methods are widely used in line drawings' matching. These techniques offer a compact representation model of the lineal image, which allows its translation, rotation and scale-invariant recognition. The structural recognition has two tendencies, sometimes equivalent, *syntactic recognition* and *graph-based recognition*.

Syntactic recognition [FWL89] is adequate when there is a limited set of models to be recognized and all of them can be described using a grammar with a reduced set of primitives. Introducing a new pattern to be recognized would imply making a grammatical inference [Fu80], and we should also suppose that this new model can be described by the same set of primitives. Works based on graph theory [LKG90], [KU86], [Hab91] allow inexact matching and rapid introduction of new patterns to be recognized without the system being altered. Besides, there is a great similarity between a graph and a line drawing. In our case, being able to perform inexact matching is essential because the input is hand drawn. For all these reasons, we are left with a method based on graph theory to attain our purposes.

Next section overviews the system. Errors due to the fact that the input document is hand drawn are characterized in section 3. Section 4 describes the node labeling process (node consistency verification). Local consistency verification (path consistency verification) of hypotheses is described in section 5. The last step of the algorithm (path consistency verification) is described in section 6. Finally, section 7 present some representative results.

2 Outline of the Approach

The output of the syntactic level is an attributed graph-based representation of the line drawing. A complete description of both lexical and syntactic levels can be found in [**?**]. The input line drawing and the patterns to be recognized are represented using a *two-level attributed graph* (*2LG*) (figure 2). A lineal drawing is represented, in the first level, by a set of attributed connected graphs. The characteristic points in the line drawing correspond to vertices of the graph and the line segments joining these points correspond to the edges. The second level graph is an hypergraph whose nodes are first level graphs and whose edges denote topological relationships between them.

Starting from this representation, *matching* is carried out using subgraph isomorphism techniques, that is, by finding the *2LG* (model graph) representing the pattern to be recognized in terms of a subgraph of the *2LG* (candidate graph) that approximates the input line drawing at best. This subgraph isomorphism process must be applied at each level of the *2LG*. The subgraph isomorphism problem is equivalent to the *consistent labeling problem* [Hen90] which is a *NP-Complete* problem. This makes any direct solution generating mechanism very sensitive to the size of model and candidate graph and requiring abusive calculation time. Hence, it becomes necessary to apply prune criteria (e.g., discrete relaxation

Figure 2: Line drawing representation by two level graphs (*2LG*).

techniques) to achieve a significant decrease in the number of possible solutions.

The consistent labeling problem may be explained as follows: O is a set of objects to be identified, L is a set of labels that represent hypotheses on the identification of the objects and R is a set of constraints between the pairs object-label. The goal is to obtain a set H of hypotheses that assigns a label to each object satisfying the existing constraints. In our case, the objects to be labeled are the model nodes. Labels are build from the candidate nodes. Therefore, the model graph can be interpreted as a constraint graph. These constraints, based on graph's edges, represent geometric and topologic conditions to be accomplished by nodes in their junctions.

From this point of view, in [Hen90] the labeling problem is decomposed in three steps:

- *Node consistency*: each node of the model graph is labeled with all possible candidate nodes matching with it. The labeling is based on topological criteria (f.i. the number of edges for which the node is an end point) and geometrical criteria (f.i. the configuration of each node, that is, the arrangement of edges for which the node is an end point).

- *Arc consistency*: the consistency of labels for each pair of neighboring nodes (linked by an edge) is checked and inconsistent labels are removed. The *AC4* (*Arc Consistency 4*) algorithm [MH86] [Hen90] is here applied. This algorithm is based on discrete relaxation techniques. These techniques have been used in previous works on line-drawing recognition [MT90] [Hab91].

- *Path Consistency*: a set of labels globally valid for all the model nodes is searched. This set of labels is considered as solution to the problem.

3 Error typification

In a vectorized document obtained from a hand drawn design there are several accuracy errors due to inaccurate stroke and to the fact that digitalization implies discretization. In this work, we propose the following error classification:

- **point duplication with touch occurrence** (Fig. 3(a)): due to an inexact stroke when there is more than one line ending at an union point. The result is the appearance in the graph of additional vertices linked by erroneous edges (see fig. 3(f)).

- **point duplication without touch occurrence** (Fig.3(b)): occurs when lines do not reach a union point, it is also due to a lack of precision in line trace. Similarly to the previous case, the effect of such error on the graph is the apparition of more vertices although now no additional edges appear (see fig.3(g)).

- **Added points** (Fig. 3(c)): when a line has been drawn using more than one stroke and little precision at linking strokes. The result of this error on the graph is the apparition of several fictitious vertices linking several edges which in fact should be a single one (see fig.3(h)).

- **Added lines** (Fig. 3(d)): are very short lines that go beyond drawing's lines and junctions. They arise when traces are not uniform, that is, when there are stains on lines or when lines have been drawn beyond the junction. In the graph short edges that should be discarded arise (see fig. 3(i)).

- **Broken lines** (Fig. 3(e)): when lines that should be solid are formed by more than one line because of a difference in stroke's pressure. The result on the graph is the apparition of two edges and two false terminal points (see fig. 3(j)).

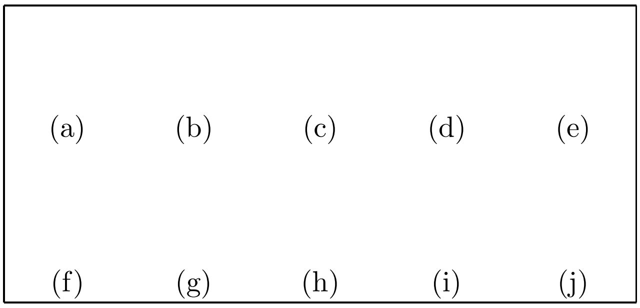

Figure 3: Accuracy errors in a hand drawn document: point duplication with (a)(f), and without touch occurrence (b)(g), added points (c)(h), added lines (d)(i), broken lines (e)(j).

Any of these errors, which may appear combined, increases exponentially the number of local hypotheses and, consequently, the necessary calculation time for isomorphism.

4 Node labeling

Notation: A graph is represented, using a standardized notation, by $G(V, A)$ where V is a set of nodes and A is a set of edges.

Given a model graph $G_M(V_M, A_M)$ and a candidate graph $G_C(V_C, A_C)$, the first step of the isomorphism assigns a set L_m of consistent labels to each node $m \in V_M$. Bearing this in mind, we have utilized the label model proposed by [Hab91]. Following this notation a label contains information on which candidate node is assigned and in which orientation. A *consistent label* $l \in L_m$ associated to a node $m \in V_M$ is defined as follows:

$$l = (n,\, w,\, el)$$

where $n \in V_C$ is a candidate node; w is the *weight* of the label where $weight = d(n) - d(m)$; $d(m)$ is the *degree* of the node m, that is, the number of edges for which m is an end point; and $el = [a_1, \ldots, a_{d(m)}]$ ($a_i \in A_C \cup \{\lambda\}$) is a circular list of the edges of the candidate node n that match with the edges of the model node m. It can be an empty edge (λ) when $w < 0$, that is, when any edge of the model edges does not appear in the candidate graph. By including empty edges inexact matching is permitted. According to this notation, a *labeling hypothesis* is a pair (m, l) where l is a consistent label for a node $m \in V_M$.

Node consistency is satisfied when there is a correspondence between angles of joining edges. If two edges of the candidate node, a and b, are compared with their corresponding edges x and y of the model node, where x, y form an angle θ, and they coincide; then the angle between a and b must be $\theta \pm \Delta$ so that the label becomes admissible in the model node. Δ is a pre-established variability margin which depends on image resolution. As you can see in figure 4, given a model node m and a candidate node n, several different consistent labeling hypotheses can be defined taking into account the different rotations of the model with respect to the candidate and the likable existence of empty edges (λ) in the label. In this figure we can see that a candidate node may match with different positions of a model node depending on edges' correspondence. According to the notation of [Hab91], in this figure, the first edge in the circular list of edges for which model node is an end point is marked with an $*$. The edges' order is taken in counterclockwise sense.

5 Local Consistency Verification

After label generation, the next step consists in verifying the consistency between labels assigned to neighboring nodes. With this aim, the *AC4* algorithm is applied. This is a discrete relaxation algorithm that runs in polynomial time and yields locally consistent solutions, i.e. arc consistency is verified. The problem of this algorithm is the great storage capacity required, however it is one of the best algorithms based on constraint propagation techniques.

All the locally consistent labels in each node of the model graph should be validated using the labels of the neighboring nodes. A binary constraint $\mathcal{R}$ between hypotheses of neighboring nodes is imposed. Let us consider a model node m_i and a label l_a assigned to it in the previous step. We will say that l_a is an *admissible label* in the node m_i, if it is consistent (according to the relationship $\mathcal{R}$) with, at least, one of the labels of any node m_j linked to m_i. When the algorithm finds a node m_j that has no compatible labels with l_a, this label is removed from the set of admissible labels for node m_i. All the hypotheses (m_j, l_b) which are in relation with (m_i, l_a) are then informed that this hypothesis is not admissible. The process is recursively repeated until stability.

The constraint $\mathcal{R}$ used can be explained as follows: given two labeling hypotheses $h1 = (m_1, (n_1, w_1, el_1))$ and $h2 = (m_2, (n_2, w_2, el_2))$, where m_1 and m_2 are joined by an edge $e \in A_M$,$\mathcal{R}(h_1, h_2)$ is true if the following constraints are satisfied:

1. There is a sequence of K edges $[b_1, \ldots, b_K]$ ($b_i \in A_C \cup \{\lambda\}, \forall i = 1 \ldots K$) which links n_1 and n_2 and which approximates the e's path.

2. Each label contains information on the rotation

Figure 4: Labeling hypotheses generated from a pair (*model node, candidate node*).

Figure 5: Example of inconsistent hypotheses.

that must be suffer the model node to match with the candidate node. This information is supplied by the circular lists of edges el_1 and el_2 and its correspondence with the edges of the respective model nodes m_1 and m_2. This rotation must be the same for h_1 and for h_2.

The example of figure 5 illustrates the constraint $\mathcal{R}$. In this example, two hypotheses are checked. The first condition is satisfied because the model nodes m_1 and m_2 are linked by the sequence $[b_4, b_5, b_6]$ that approximates the model edge a_1 (straight line). However, the second condition is not satisfied because in the label h_1, the model edge a_1 corresponds to the candidate edge b_4 and, in the label h_2, a_1 corresponds to candidate edge b_3, i.e. h_1 and h_2 denote different rotation for the model nodes m_1 and m_2.

6 Global Consistency Verification

Although discrete relaxation techniques find a global solution sometimes, they do not always, but they reduce the search space of the labeling problem that is an exponential problem. The last step of the matching process consists in finding a set of hypotheses, one for each model node, constituting a globally valid solution. In this step the path consistency is verified, that is, if there is a path between two model nodes and all of the hypotheses corresponding to nodes of the path must be compatible.

The global consistency verification is based on a depth first search algorithm. In the tree search, each level represents a model node and each node represents a label associated to that model node. We have a solution when we reach a leaf in the tree search. A global solution is a set of labeling hypotheses $H = \{h = (m, l), \forall m \in V_M\}$ where, if there exists an edge $e \in A_M$ that links two hypotheses $h_i, h_j \in H$, then $\mathcal{R}(h_i, h_j)$ or $\mathcal{R}(h_k, h_j)$ or $\mathcal{R}(h_i, h_l)$ is true. h_k is a h_i's matching hypothesis and h_l is a h_j's matching hypothesis. The *matching hypotheses* criterion is introduced to solve some problems as the one in figure 6. In this example, due to one of the errors characterized in section 3, a model node m has been properly labeled for two candidate nodes as a result of the duplication of the characteristic point. The hypotheses h_1 and h_2 are two valid interpretations for the global consistency verification but, if we consider them independently, none of them turns out to be globally valid. Both hypotheses have complementary information that must be taken into account. The hypotheses h_1 and h_2 match because they have the same model node n, the candidate nodes n_1 and n_2 are neighbors and if we match the h_1's edges circular list with the h_2's edges circular list, the angles between edges of m are satisfied.

Figure 6: Example of matching hypotheses.

7 Experimental results

(a) (b) (c) (d)

Figure 7: Some model graphs to be matched.

This section illustrates the algorithm described above with some meaningful examples. In figure 7 some models to be found in the image are shown. Figure 8(a) is the original image corresponding to the scanned document. Figure 8(b) shows the *2LG* approximation obtained as a output of the syntactic level. In this figure we can realize that the segments of the original image have been approximated by straight lines and circular arcs. Similarly, because input documents are hand drawn, some of the errors described in section 3 occur in this figure once more. Figures 8(c), 8(d), 8(e) and 8(f) are respectively the results of subgraph matching with the patterns of figures 7(a), 7(b), 7(c) and 7(d). In these figures, the edges corresponding to a solution have been bolded. All of the results show that inexact matching has been here performed. In figures 8(c) and 8(d) some problems are solved: the splitting of the characteristic points and the approximation of a model edge by means of a sequence of candidate edges. Figures 8(e) and 8(f) show the result of matching using patterns that contain circular arcs.

8 Conclusion

This paper has presented a method to recognize architectural documents using discrete relaxation techniques. The algorithm described should be seen as a step in the interpretation of the drawing. This interpretation allows converting a paper-based document into an electronic document and integrating it in a CAD system. A graph-based model has been chosen to represent the input documents. The AC4 algorithm has been implemented to make the matching step. The main contribution of this work is to start from hand drawn plans, in which it is necessary to identify accuracy errors. These errors have been taken into account in the development of the matching algorithm. A coherence constraint for the local consistency verification and a matching hypothesis criterion have been proposed to allow an inexact matching. Using graph-based techniques allows a translation, rotation and scale-invariant recognition and an easy integration of new symbols to recognize. When all patterns have been recognized, a high level representation of the document based on a semantic net representation is built. The method developed in this paper has been used for architectural drawings, however, it's a general method that can be easily adapted to other domains.

References

[BCDB*92] Luca Boatto, Vincenzo Consorti, Monica Del Buono, Silvano Di Zenzo, Vinzenzo Eramo, Alessandra Espossito, Francesco Melcarne, Marco Meucci, Andrea Morelli, Marco Mosciatti, Stefano Scarci, and Marco Tucci. An interpretation system for land register maps. *Computer*, 25(7):25–33, July 1992.

[Ble84] Heinrich Bley. Segmentation and preprocessing of electrical schematics using picture graphs. *Computer Vision, Graphics, and Image Processing*, 28(28):271–288, 1984.

[BWS88] J.P. Bixler, L.T. Watson, and J.P. Sanford. Spline-based recognition of straight lines and curves in engineering line drawings. *Image and Vision Computing*, 6(4):262–269, november 1988.

[Cha90] Shi-Kuo Chang. *Principles of pictorial information systems design.* Prentice-Hall, 1990.

[Fu80] K.S. Fu. *Digital Pattern Recognition.* Springer-Verlag Berlin Heidelberg, 1980.

[FWL89] C.S. Fahn, J.F. Wang, and J.Y. Lee. A topology-based component extractor for understanding electronic circuit diagrams. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(11):1140–1157, 1989.

[Hab91] A.H. Habacha. Structural recognition of disturbed symbols using discrete relaxation. In *1st. ICDAR*, pages 170–178,

(a) (b)

(c) (d)

(e) (f)

Figure 8: Original image (a), graph image (b), isomorphism with patterns (c)(d)(e)(f).

September-October 1991. Saint Malo, France.

[Hen90] Thomas C. Henderson. *Discrete Relaxation Techniques*. Oxford University Press, 1990.

[KS93] Lih-Shing Kung and Jean-Claude Samin. A procedure of reading mechanical engineering drawings for cad applications. *Signal Processing*, (32):191–200, 1993.

[KU86] P. Kuner and B. Ueberreiter. Knowledge-based pattern recognition in disturbed line images using graph theory, optimization and predicate calculus. In *8th. ICPR*, pages 240–243, October 1986. Paris, France.

[LKG90] S.W. Lee, J.H. Kim, and F.C.A. Groen. Translation-, rotation-, and scale-invariant recognition of hand-drawn symbols in schematic diagrams. *International Journal of Pattern Recognition and Artificial Intelligence*, 4(1):1–25, 1990.

[Mad91] D. Madej. An intelligent map-to-cad conversion system. In *1st. ICDAR*, pages 602–610, September-October 1991. Saint Malo, France.

[MH86] R. Mohr and T.C. Henderson. Arc and path consistency revisited. *Artificial Intelligence*, (28):225–233, 1986.

[MT90] G. Masini and K. Tombre. Discrete relaxation applied to interpretation of technical documents. In *10th. ICPR*, pages 706–708, June 1990. New Jersey, USA.

[NL90] V. Nagasamy and N.A. Langrana. Engineering drawing processing and vectorisation system. *Computer Vision, Graphics and Image Processing*, (49):379–397, 1990.

[PLJ91] D. Pao, H. F. Li, and R. Jayakumar. Graphic features extraction for automatic conversion of engineering line drawings. In *1st. ICDAR*, pages 533–541, September-October 1991. Saint Malo, France.

[SH89] Joseph S.H. Processing of enginnering line drawings for automatic input to cad. *Pattern Recognition*, 22(1):1–11, 1989.

[ST82] T. Sato and A. Tojo. Recognition and understanding of hand-drawn diagrams. *IEEE CH1801-0/82/0000/0674*, 1982.