Machine Vision and Applications (1997) 10:150–158

Machine Vision and Applications
© Springer-Verlag 1997

A system to understand hand-drawn floor plans using subgraph isomorphism and Hough transform

Josep Lladós[1], Jaime López-Krahe[2], Enric Martí[1]

[1] Computer Vision Center – Dep. Informàtica. Universitat Autònoma de Barcelona, 08193 Bellaterra (Barcelona), Spain
e-mail: josep@cvc.uab.es, enric@cvc.uab.es
[2] Dep. Informatique, Lab. ai/mime, Univ. Paris 8, Saint Denis, 93526 Paris CEDEX 02, France
e-mail: lopez@ai.univ-paris8.fr

Received 6 June 1996 / Accepted 4 February 1997

Abstract. Presently, man-machine interface development is a widespread research activity. A system to understand hand drawn architectural drawings in a CAD environment is presented in this paper. To understand a document, we have to identify its building elements and their structural properties. An attributed graph structure is chosen as a symbolic representation of the input document and the patterns to recognize in it. An inexact subgraph isomorphism procedure using relaxation labeling techniques is performed. In this paper we focus on how to speed up the matching. There is a building element, the walls, characterized by a hatching pattern. Using a straight line Hough transform (SLHT)-based method, we recognize this pattern, characterized by parallel straight lines, and remove from the input graph the edges belonging to this pattern. The isomorphism is then applied to the remainder of the input graph. When all the building elements have been recognized, the document is redrawn, correcting the inaccurate strokes obtained from a hand-drawn input.

Key words: Line drawings – Hough transform – Graph matching – CAD systems – Graphics recognition

1 Introduction

CAD systems are of great help to create and modify technical documents efficiently. But, what about the reverse problem, converting paper-based drawings for their integration into a CAD environment? The field of *document image analysis* gives a positive answer to the question thanks to image processing and pattern recognition techniques applied to scanned images of documents. Here, we focus on hand-drawn floor plans for which we propose an alternative CAD system input technique. 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 quick and easy manner.

Correspondence to: J. Lladós

In this paper we present a system to understand hand-drawn floor plans. Understanding a plan consists of recognizing building elements (doors, windows, walls, tables, etc.) and their topological properties. The input document is scanned and vectorized. The vectorization module (Lladós et al. 1993) generates an attributed graph representation of the drawing. We can distinguish two kinds of structural elements to recognize: those that have a fixed pattern and those that can be recognized by their textural properties. The models belonging to the first set are also represented with an attributed graph structure, and a graph-matching process is performed to do the recognition. The elements of the second type are characterized by a filling texture and, to recognize them, we will search in the input graph for the features of this texture by means of a Hough-based technique.

Attributed relational graphs have been widely used to represent and recognize line drawings. Recognition is performed using graph-matching procedures that find a subgraph isomorphism between a model graph and an input graph representing the input document. Some outstanding examples of graph-matching techniques applied to line drawing recognition can be found in the literature (e.g. Kuner and Uberreiter 1988; Habacha 1991; Lee et al. 1990). The subgraph isomorphism problem has been classically solved by backtracking tree search procedures but, since it falls into the NP-complete complexity class, some heuristic techniques have been proposed to prune the search space and speed up the matching. Discrete relaxation (Henderson 1990) is a constraint propagation technique that allows the removal of inconsistent hypotheses before tree search expansion. Besides the exponential computational load required by graph matching, another obstacle is to deal with disturbed graphs obtained from noisy data. In hand-drawn documents, this problem is clearly noticeable because of the uncertainty induced by hand-drawn strokes. To solve this drawback, an *inexact graph-matching* process has to be implemented, introducing a modelization of graph distortion. So, an inexact graph-matching procedure will look for the best graph that matches the model graph, i.e. the graph representing the minimal deformation regarding the model graph. The best known algorithm for inexact graph-matching is an extension of the model used in string edit distance (Wagner and

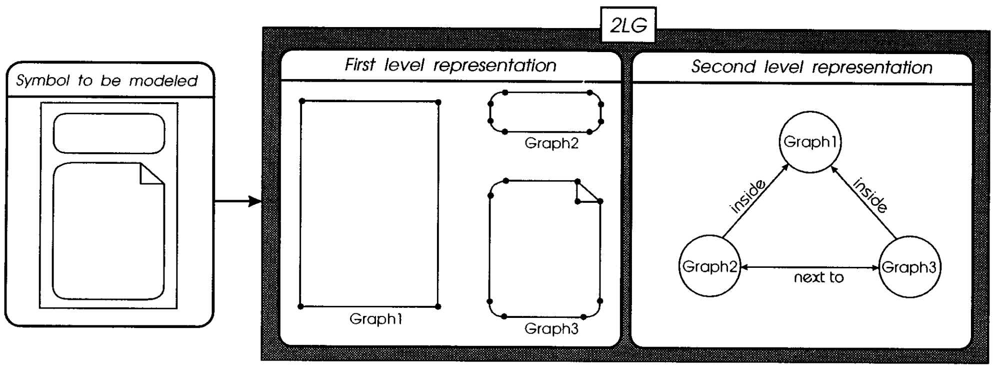


Fig. 1. Line drawing representation by two level graphs (*2LG*)

Fisher 1974). It is based on the idea of introducing graph edit operations and computing the cheapest set of modifications required to transform one graph into another. With an appropriate definition of the compatibility constraints, relaxation labeling allows also to search for an inexact matching. In our case, the matching process uses the *AC4* algorithm (Mohr and Henderson 1986) based on discrete relaxation techniques.

The second type of structures to recognize in the linear drawing are those that do not have a fixed pattern, thus a matching process cannot be used, but are characterized by a filling texture. The problem of hatching pattern detection is an important concern in the field of document analysis (e.g. Kasturi et al. 1990; Boatto et al. 1992; Antoine et al. 1992). An efficient extraction of hatched areas allows to drastically reduce the information and to focus further interpretation on the remaining lines. Hatched area detection is often based on *a priori* knowledge about their meaning within the document (buildings, solid regions, walls, etc.) or attributes of their lines (e.g. slope or frequency). In this work, hatched areas represent walls and their attributes are extracted from the document itself. The only knowledge is that they consist of parallel even-spaced straight lines of any slope. We find these structures using a Hough-based recognition process that searches for some structural properties. Straight line Hough transform (SLHT) has been often used to understand linear images. Some characteristic configurations in the original image (parallel edges, cross points, etc.) can be easily detected in the Hough space. López-Krahe and Pousset (1988) used SLHT to detect parallel straight lines in small-scale images. Wahl (1989) detected cluster patterns in Hough space to carry out an interpretation of 3D polyhedral scenes. Pao et al. (1992) used an SLHT-based method to match continuous closed smooth curves. In our case, there is a building element, the walls, characterized by a hatching pattern. We can search for the structural features of the walls in the input graph, using an SLHT-based method to detect which graph edges belong to the textured areas filling walls. The detection starts by transforming each straight graph edge to a parameter space. The peaks in this parameter space are detected by means of a clustering process. These peaks correspond to parameter values that define the hatched areas. Moreover, this hatching pattern recognition step can also be seen as a previous filter for the graph-matching process that allows to remove several edges of the input graph. The edge removal reduces search space meaningfully and thus speeds up model matching.

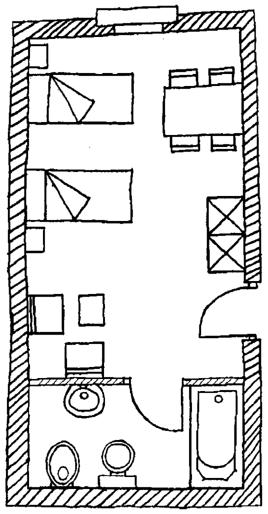
Section 2 gives a summary of the graph isomorphism procedure to detect patterns within a document. In Sect. 3, we explain how to improve the recognition by an SLHT-based method that finds textured areas. Section 4 shows the synthetized image after parameter estimation. Section 5 reports a quantitative analysis of different examples. Section 6 is devoted to the conclusion.

2 Overview of the graph-matching algorithm

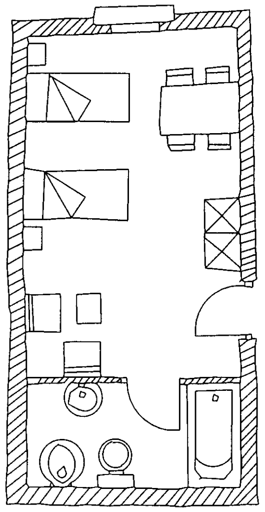
The input line drawing and the patterns to recognize are represented using a *two-level attributed graph* (*2LG*). The line information of the input image is represented, in the first level, by a set of attributed undirected graphs, one for each connected component. The nodes of each graph represent the characteristic points (junctions or end points of lines). The attributes are their position, degree (number of lines joining in the node) and the angles between these lines. The edges of the graph represent the segments joining at characteristic points. The attributes are the length and, depending on whether the segment is adjusted by a straight line or a circumference arc, the parameters that characterize the respective equation. The second-level graph is a hypergraph whose nodes are first-level graphs and whose edges denote topological relationships between them. See in Fig. 1 an example of a line drawing representation by a *2LG*.

In this paper, we will work with first-level graphs and represent them, using a standarized notation, by $G(V, E)$, where V is a set of nodes and E is a set of edges. Let $v \in V$ and $e \in E$ be a graph node and a graph edge, respectively. We will denote their corresponding attributes as: x_v, y_v (node position), d_v (node degree), $[\gamma_v^1, \ldots, \gamma_v^{d_v}]$ (angles between the lines joining at v), m_e (edge length) and θ_e, ρ_e, if e is a straight line with equation $\rho_e = xcos\theta_e + ysin\theta_e$, or xc_e, yc_e, r_e, if e is a circumference arc with equation $r_e^2 = (x - xc_e)^2 + (y - yc_e)^2$. is a circumference arc with equation $e_r^2 = (y - e_{yc})^2$. Figure 2a shows an input image to be recognized and Fig. 2b shows its *2LG* representation. We can see some differences between the input image and its graph-based representation due to the input segments approximation by straight lines and arcs.

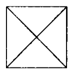
Starting from this representation, matching is carried out using subgraph isomorphism techniques, that is, finding the



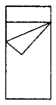
2a



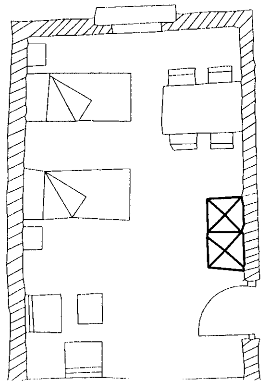
2b



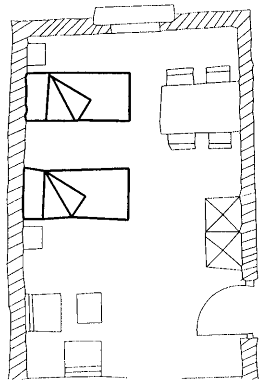
3a



3b



3c



3d

Fig. 2. a Input image. **b** *2LG* representation after vectorization

Fig. 3. a, b Some model graphs to be matched; **c, d** Isomorphism against patterns

2LG (model graph) representing the pattern to be recognized in terms of a subgraph of the *2LG* (input graph) that approximates the input line drawing at best. The subgraph isomorphism problem is equivalent to the *consistent labeling problem* (Henderson 1990) which 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 and the labels are built from the input nodes. Therefore, the model graph can be interpreted as a constraint graph. These constraints, based on the graph's edges, represent geometric and topologic constraints to be accomplished by nodes in their junctions. Subgraph isomorphism is solved using the *AC4* algorithm. Given a model graph $G_{\mathrm{M}}(V_{\mathrm{M}}, E_{\mathrm{M}})$ and the input graph $G_{\mathrm{I}}(V_{\mathrm{I}}, E_{\mathrm{I}})$, the algorithm runs in two steps:

1. **Node labeling**. A set L_v of consistent labels is assigned to each node $v \in V_{\mathrm{M}}$ according to their local configuration, i.e. the edges joining at these nodes and the angles between them. A *consistent label* $l \in L_v$ associated to a node $v \in V_{\mathrm{M}}$ is defined as $l = (w, el)$, where $w \in V_{\mathrm{I}}$ is an input node; and $el = [e_1, \ldots, e_{d_v}]$ ($e_i \in E_{\mathrm{I}} \cup \{\lambda\}$) is a circular list of edges joining at w, which match, following a counterclockwise order, with the edges joining at v. Some e_i may be an empty edge (λ) when there is any input edge matching the corresponding model edge. According to this notation, a *labeling hypothesis* is a pair (v, l) where l is a consistent label for a node $v \in V_{\mathrm{M}}$.
2. **Arc consistency verification**. All the locally consistent labels in each model node should be validated using the labels of the neighboring nodes. A constraint $\mathscr{R}$ between hypotheses of neighboring nodes is imposed. In our case, the constraint $\mathscr{R}$ can be explained as follows: given two labeling hypotheses $h1 = (v_1, (w_1, el_1))$ and $h2 = (v_2, (w_2, el_2))$, where v_1 and v_2 are joined by an edge $e \in E_{\mathrm{M}}$, $\mathscr{R}(h_1, h_2)$ is true if 1) there is a sequence of K edges $[b_1, \ldots, b_K]$ ($b_i \in E_{\mathrm{I}} \cup \{\lambda\}, \forall i = 1 \ldots K$) which links v_1 and v_2 and which approximates the e's path, and 2) h_1 and h_2 denote the same rotation for the model nodes v_1 and v_2.

Inexact graph-matching is attained by the inclusion of empty edges and allowing some distortion when model nodes and input nodes are compared. Figure 3c, d are the results of inexact subgraph-matching with the patterns of Fig. 3a, b, respectively. In these figures, the edges corresponding to a solution have been bolded. See Lladós and Martí (1995) for further details on the isomorphism algorithm.

3 Hatching patterns recognition

Wall detection involves two purposes at the same time: recognizing a building element which does not have a pattern graph and pruning the tree search before starting subgraph isomorphism to speed up this process. Wall recognition will be carried out by detection of textured areas filled with parallel even-spaced straight lines. We will make two additional assumptions for these regions: there are two directions allowed for the walls and these two directions must be orthogonal. It is possible to relax these assumptions and allow more than two directions. In this case, since the detection of dominant directions involves a supervised learning process, the number of directions should be an input parameter to the system.

Classical SLHT transforms each image point (x, y) into a set of points (θ, ρ) that fulfill the equation $\rho = xcos\theta + ysin\theta$. Detection of peaks in the parameter space constitutes a powerful method to detect straight lines in the input image. Several applications have been developed from this idea. See a complete survey in Illingworth and Kittler (1988) and Leavers (1993). With the aim to detect textured regions, we use the idea of SLHT in the following definition:

Definition 1. Given an attributed graph $G(V, E)$, we define its *graph-based Hough transform* with parameters θ and ρ as a function $\mathrm{GBHT}_{\theta\rho} : E \to [0,\pi] \times \mathbb{R}$ that, for each straight edge $e \in E$ with attributes values θ_e and ρ_e, transforms e into a point (θ_e, ρ_e) in the θ-ρ parameter space. If the considered attributes of e are θ_e and its length m_e, we define analogously $\mathrm{GBHT}_{\theta m}$ in θ-m parameter space.

Figure 4 shows these two possible transformations in an ideal case. $\mathrm{GBHT}_{\theta\rho}$ gives, for each wall, two peaks with the same θ and a difference in ρ equal to the wall's width. $\mathrm{GBHT}_{\theta\rho}$ also gives a sequence of points with the same θ and even spaced. With this configuration we can calculate the orthogonal dominant directions, the filling edges direction and the wall's width. On the other hand, $\mathrm{GBHT}_{\theta m}$ allows to calculate the same parameters but in a more accurate manner. $\mathrm{GBHT}_{\theta m}$ gives three peaks in the parameter space. Two of them have the same m and a difference of $\pi/2$ in θ and correspond to the orthogonal dominant directions. The third peak corresponds to the filling edges, and the wall's width (L_1 in the figure) is calculable from it, as can be seen in the next section. The inaccuracy of hand-drawn design causes that graph edges with the same a priori attribute values will be slightly different after vectorization. This fact provokes scattered points in Hough space and additional difficulty in the detection of peaks. For this reason, we have used $\mathrm{GBHT}_{\theta m}$, which is more robust because the information required is concentrated only in three peaks. The properties of the walls are extracted from the θ-m space. Then, the size of the input graph is reduced and the subgraph isomorphism against the models is done with the remainder of input graph, i.e. input graph after removing its edges belonging to hatched regions. Introducing this Hough-based procedure, the algorithm of document interpretation and reconstruction runs in five steps: it starts by $\mathrm{GBHT}_{\theta m}$ computation of the input graph. Then, some structural features of walls are calculated from peaks in θ-m space. The next step is the extraction of vertical and horizontal connected components that will be defined as subgraphs of the input graph which outlines the walls. With all this information, a new graph structure that stores wall features is built. The final step carries out an inexact matching between the remainder of input graph and the model graphs of building elements. These steps are described in the following sections.

3.1 Parameter computing

A clustering process is applied over θ-m space, obtained as output of $\mathrm{GBHT}_{\theta m}$, with the aim to detect the three peaks mentioned above. Two of these three peaks should have a difference of $\pi/2$ and correspond to orthogonal dominant directions. The third peak is used to calculate the filling direction and the wall's width. The well-known *k-means* algorithm is used to classify the points of θ-m space in three classes, C_V, C_H and C_F. The centers of these three classes are taken as the three peaks shown in an ideal case in Fig. 4. Figure 5a shows θ-m space after $\mathrm{GBHT}_{\theta m}$ of Fig. 2b input graph. Figure 5b shows θ-m space smoothed with a Gaussian filter. We have stated experimentally that the algorithm works faster and gives better results if k-means is initialized from the maxima of smoothed θ-m space. Peaks in the above image can be better assessed in Fig. 5c which shows a 3D representation of θ-m space. Notice that, since the θ axis is cyclic, in the clustering process, the distance between two points must be defined in terms of a cyclic distance. On the other hand, it can be seen that the cluster corresponding to filling edges, characterized by its center $(\theta_\mathrm{F}, m_\mathrm{F})$ in Fig. 5d, actually contains two peaks. This is because, see Fig. 2, the input document contains walls with two different widths. This fact results in a cluster containing two near peaks and, thus, the computed centre $(\theta_\mathrm{F}, m_\mathrm{F})$ represents an average of them. As will be seen, the more scattered the filling cluster, the larger the computed range of variation for wall width. Finally, Fig. 5d shows the three classes found with their corresponding dominant directions θ_V, θ_H and θ_F and the average length m_F of filling edges. Let (θ_1, m_1), (θ_2, m_2) and (θ_3, m_3) be the three peaks detected in θ-m space. We can calculate the following information:

– **Dominant directions.** θ_1, θ_2 and θ_3 are the angles corresponding to the three dominant directions of the input edges. The first step is to find, out of θ_1, θ_2 and θ_3, which two directions are orthogonal. Let θ_{o1} and θ_{o2} denote the angles corresponding to these two directions. The third direction will be the filling direction and its angle will be denoted as θ_F.
– **Graph rotation value.** Let V_α and H_α be the minimal rotation angle which must be applied to a direction with angle α to align it respectively with vertical and horizontal directions. The rotation θ_rot that must be applied to the input graph to align its dominant directions with vertical and horizontal directions can be calculated as follows:

$$\theta_\mathrm{rot} = min(\frac{1}{2}(V_{\theta_{o1}} + H_{\theta_{o2}}), \frac{1}{2}(H_{\theta_{o1}} + V_{\theta_{o2}}))\,. \qquad (1)$$

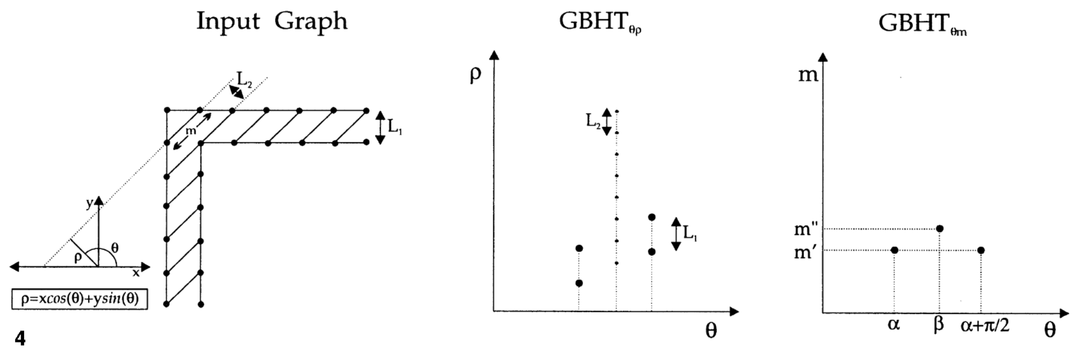
If we assume $|\theta_{o1} - \theta_{o2}| = \pi/2 \pm \delta$, then the following equalities must also be satisfied: $|V_{\theta_{o1}} - H_{\theta_{o2}}| = \delta$ and $|H_{\theta_{o1}} - V_{\theta_{o2}}| = \delta$. After finding θ_rot, we can know which direction, θ_{o1} or θ_{o2}, corresponds to vertical direction and which one corresponds to horizontal direction. Let θ_V and θ_H be respectively the dominant directions closest to vertical and horizontal directions.
– **Direction variation.** Let $\varDelta_\mathrm{H}$, $\varDelta_\mathrm{V}$ and $\varDelta_\mathrm{F}$ be the range of variation allowed for the dominant directions.

Definition 2. Given an input graph $G_\mathrm{I}(V_\mathrm{I}, E_\mathrm{I})$, we define the *set of vertical edges* $E_\mathrm{V} \subseteq E_\mathrm{I}$ as $E_\mathrm{V} = \{e \in E_\mathrm{I}, \theta_e = \theta_\mathrm{V} \pm \varDelta_\mathrm{V}\}$. Where θ_e is the orientation of input edge e. Similarly, we can define E_H as the *set of horizontal edges* and E_F as the *set of filling edges*.

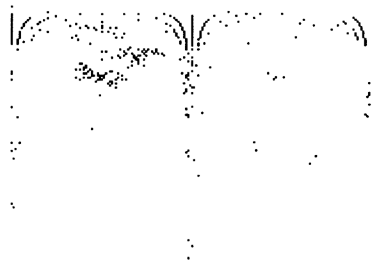
The ranges of variation $\varDelta_\mathrm{V}$, $\varDelta_\mathrm{H}$ and $\varDelta_\mathrm{F}$ can be calculated from the average deviation on θ of each class C_i obtained after the clustering process. So, given a dominant direction θ_i, its range of variation is calculated as follows:

$$\varDelta_i = \frac{1}{card(C_i)} \sum_{p \in C_i} |\theta_p - \theta_i|\,, \qquad (2)$$

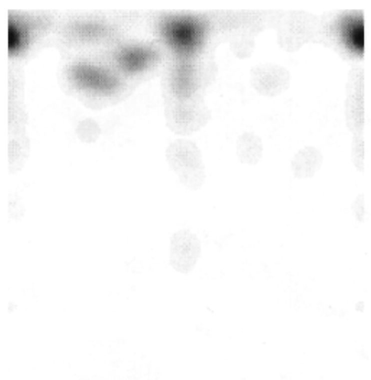
where C_i is the class whose centre is the dominant direction θ_i.
– **Wall width.** Let $(\theta_\mathrm{F}, m_\mathrm{F})$ be the class centre corresponding to filling edges. It is used to calculate the wall width.



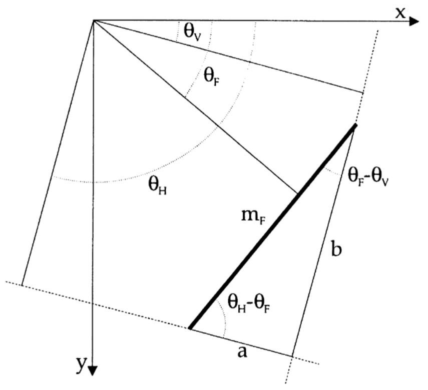
4



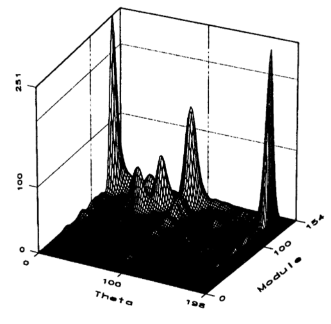
5a



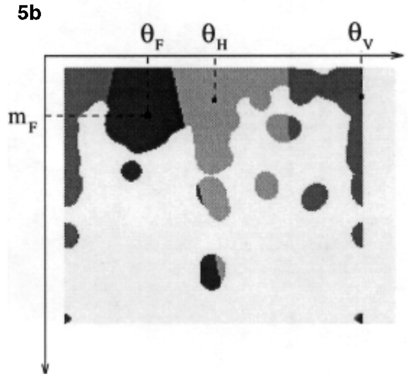
5b



6



5c



5d

Fig. 4. GBHT$_{\theta\rho}$ and GBHT$_{\theta m}$

Fig. 5. a GBHT$_{\theta m}$ obtained from the input graph. **b** GBHT$_{\theta m}$ smoothed using a Gaussian filter. **c** 3D view of the above image. **d** Clustering of θ-m space using the k-means algorithm

Fig. 6. Average wall width calculation. A wall filled by an edge with length m_F has width a or b, depending on whether it is a vertical wall or a horizontal wall

Given an ideal filling edge with length m_F and orientation θ_F (see Fig. 6), the width of the wall containing this edge will be a or b, depending on whether it joins vertical or horizontal edges, respectively. These values can be estimated according to the following equations:

$$a = m_F \cos(\theta_H - \theta_F) = m_F \sin(\theta_F - \theta_V)\,, \tag{3}$$

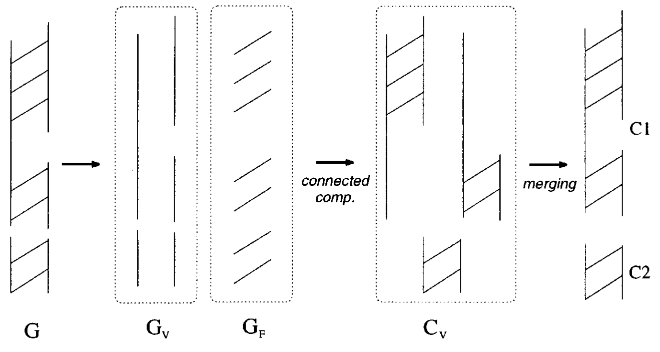
$$b = m_F \sin(\theta_H - \theta_F) = m_F \cos(\theta_F - \theta_V)\,. \tag{4}$$

A filling edge, after GBHT$_{\theta m}$, is transformed into a point (θ_F, m_F) in the parameter space. In the estimation of average wall width we cannot know which value, a or b, must be taken, because both types of filling edges, vertical and horizontal, contribute in the accumulation point (θ_F, m_F). We estimate an interval $[W_{\min}, W_{\max}]$ for wall width depending on a and b values and the average deviation of filling edge length. Since a large interval $[W_{\min}, W_{\max}]$ is obtained when the filling cluster contains peaks corresponding to walls of different widths, a more accurate computation of each wall width is performed after its extraction. This interval is calculated as follows:

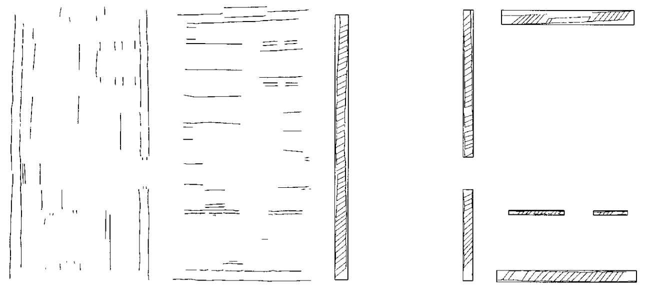
$$W_{\max} = (m_F + \Delta_F)\, max(\cos\beta, \sin\beta)\,, \tag{5}$$

$$W_{\min} = (m_F - \Delta_F)\, min(\cos\beta, \sin\beta)\,, \tag{6}$$

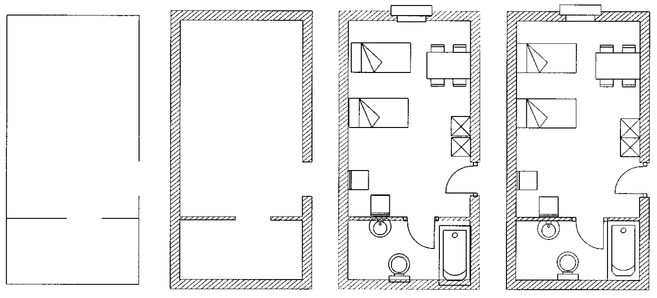
where $\beta = \theta_H - \theta_F$ and Δ_F is the average deviation in m corresponding to the filling edges class in the θ-m space calculated as follows:



7



8a 8b 8c 8d



9a 9b 9c 9d

Fig. 7. Vertical connected components extraction and merging. Given a graph G, we extract first its subgraph of vertical edges (G_{V}) and its subgraph of filling edges (G_{F}). Combining these two graphs, a set C_{V} with three vertical connected components is created. Two connected componets are merged when they have a common vertical chain

Fig. 8. **a** Subgraph of vertical edges $G_{\mathrm{V}}(V_{\mathrm{V}}, E_{\mathrm{V}})$. **b** Subgraph of horizontal edges $G_{\mathrm{H}}(V_{\mathrm{H}}, E_{\mathrm{H}})$. **c** Vertical connected components and their bounding boxes. **d** Horizontal connected components and their bounding boxes

Fig. 9. **a** Walls graph; **b** walls reconstruction; **c** document reconstruction after subgraph isomorphism against some models; **d** result after overlapping correction

$$\Delta_{\mathrm{F}} = \frac{1}{card(C_{\mathrm{F}})} \sum_{p \in C_{\mathrm{F}}} |m_p - m_{\mathrm{F}}| \,. \qquad (7)$$

3.2 Connected component extraction

After walls parameter estimation, we filter the input graph to obtain E_{V}, E_{H} and E_{F}. These edge sets are used to define three subgraphs of input graph $G_{\mathrm{I}}(V_{\mathrm{I}}, E_{\mathrm{I}})$:

Definition 3. We define $G_{\mathrm{V}}(V_{\mathrm{V}}, E_{\mathrm{V}})$ as the *subgraph of vertical edges of* G_{I}. E_{V} was defined in definition 2 and V_{V} is defined as $V_{\mathrm{V}} = \{v \in V_{\mathrm{I}}, \exists e = (v_1, v_2) \in E_{\mathrm{V}}, v = v_1$ or $v = v_2\}$. In the same manner, we define the subgraphs $G_{\mathrm{H}}(V_{\mathrm{H}}, E_{\mathrm{H}})$ and $G_{\mathrm{F}}(V_{\mathrm{F}}, E_{\mathrm{F}})$, using horizontal edges and filling edges, respectively.

G_{V} and G_{H}, shown in Fig. 8a and b, respectively, do not contain enough information to extract vertical and horizontal walls, because other picture elements contain vertical and horizontal edges, too. It is necessary to search for pairs of vertical or horizontal edges joined by a filling edge. This is the basic idea to build vertical and horizontal connected components, defined according to the following definitions:

Definition 4. A *vertical chain* $L_{\mathrm{VC}}(V_{\mathrm{VC}}, E_{\mathrm{VC}})$ is defined as a connected subgraph of G_{V} such that its vertex set V_{VC} can be ordered in a sequence $[v_{i_1}, v_{i_2}, \ldots, v_{i_I}]$ such that, for all $j = 2 \ldots I$, $(v_{i_{j-1}}, v_{i_j}) \in E_{\mathrm{VC}}$.

Definition 5. Given $G_{\mathrm{V}}(V_{\mathrm{V}}, E_{\mathrm{V}})$ defined according to definition 3, a *vertical connected component* is defined as a 3-tuple $C_{\mathrm{V}} = \langle L_1, L_{\mathrm{F}}, L_2 \rangle$, where L_1 and L_2 are vertical chains and L_{F} is a subgraph of G_{F} whose edges join L_1 and L_2. A *horizontal connected component* is accordingly defined starting from G_{H}.

According to definition 5, we will denote C_{V} (C_{H}) as the set of vertical (horizontal) connected components extracted from G_{V} (G_{H}).

Given two vertical connected components $C_{\mathrm{V}_i} = \langle L_{1_i}, L_{\mathrm{F}_i}, L_{2_i} \rangle$ and $C_{\mathrm{V}_j} = \langle L_{1_j}, L_{\mathrm{F}_j}, L_{2_j} \rangle$, if $L_{1_i} = L_{1_j}$ or $L_{2_i} = L_{2_j}$, then C_{V_i} and C_{V_j} are merged in $\langle L_{1_i}, L_{\mathrm{F}_i} \cup L_{\mathrm{F}_j}, L_{2_i} \cup L_{2_j} \rangle$ or $\langle L_{1_i} \cup L_{1_j}, L_{\mathrm{F}_i} \cup L_{\mathrm{F}_j}, L_{2_j} \rangle$, respectively. The connected components merging process is iteratively repeated until stability. Figure 7 shows an example of vertical connected components extraction and merging. Horizontal connected components are also merged following the same idea.

Figure 8c and d show vertical and horizontal connected componets and their bounding boxes which constitute a first approximation of walls. For each box, we calculate the following features which characterize its corresponding wall: width, filling edge orientation and filling edge frequency. Some of these attributes had already been calculated, but we obtained them from a voting space with information on all the edges in the input graph. Now we use only the edges inside the wall to obtain these attributes, therefore it is possible to adjust them better.

4 Document redrawing and correction

A new attributed graph structure is created to represent the walls. In this graph, the edges correspond to the wall bounding boxes and the vertices correspond to the junctions between these boxes. Figure 9a shows this graph and Fig. 9b shows the wall reconstruction, starting from the above graph and its attributes.

After wall recognition, edges belonging to walls are removed from the input graph and a matching between the remainder of the input graph and the set of model graphs is carried out. After finding the models in the input graph, we store only the position, scale and orientation for each

Table 1. Quantitative analysis

Example	Image size	Input graph nodes	edges	size	Remainder graph nodes	edges	Reduction rate in matching speed
1	400 × 780 (305 Kb)	471	676	53 Kb	349	418	75%
2	850 × 610 (506 Kb)	685	971	128 Kb	578	813	58%
3	600 × 700 (411 Kb)	182	245	19 Kb	135	162	57%

instance, calculating these parameters by means of an alignment between the model nodes and their corresponding input nodes. The matching process has assigned an input node $w_j \in V_{\mathrm{I}}$ to each model node $v_i \in V_{\mathrm{M}}$. Let v_c be the centre of model nodes and w_c the centre of their corresponding input nodes. The pairs (v_c, v_i) and (w_c, w_j) define a translation, rotation and scale that align v_i with w_j. Since the matched subgraph is a distorted version of the model graph, each individual alignment between a model node v_i with its corresponding input node w_j is likely to be slightly different from the others. Thus, the mean translation, rotation and scale among individual alignments define the position, orientation and scale of each instance of model graph.

The document can be reconstructed from the wall graph, instantiating each model according to its attributes (see Fig. 9c to see the result of this reconstruction). Notice that some elements appear overlapped, because this image is a set of instances of the model graphs. The last step is a correction of element position to avoid overlapping. This correction is done by computing the overlapping area between elements and translating them until this overlapping area is empty. The result after this correction is shown in Fig. 9d.

5 Results and discussion

The process described in the previous sections has been applied to a test set containing seven hand-drawn sketches of floor plans. In addition to the results displayed in Fig. 9d, two examples from the test set have been chosen for detailed study. In this section, Figs. 2a, 10a and 11a will be referred to, as example 1, 2 and 3, respectively. Table 1 reports some data regarding the graphs extracted from these three examples, the remainder graphs after removing edges belonging to textured areas, and the rate of reduction in the processing time, depending on whether the matching is done with the input graph or the remainder graph without walls edges.

Each document has been scanned, in a range between 120 and 250 dpi, vectorized and represented by *2LG*. This representation allows to reduce the storage requirements (see the rate of this reduction in Table 1). Afterwards, the hatching pattern recognition process described in Sect. 3 was applied. Table 2 displays the parameters stated in Sect. 3 and computed for each example. Notice that, in the third example (Fig. 11), the range of variation for the three dominant directions is significatively smaller than in the previous examples. This is because the first two graphs are larger than the third one and they have a number of edges not belonging to textured regions, which disturb the clustering of the parameter space. On the other hand, we can see that filling direction (θ_{F}) is not restricted and can vary across documents. However, the method proposed in this paper is able to find this direction.

Table 2. Parameters defining walls

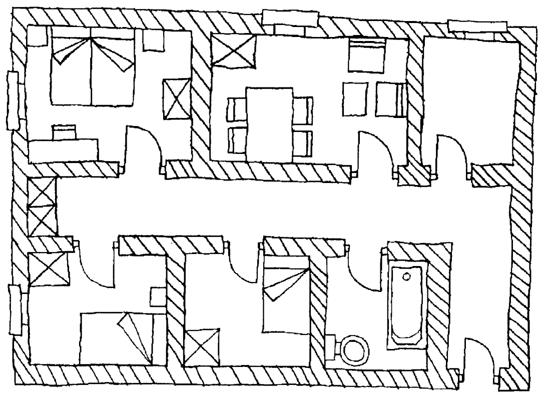
Example	$\theta_{\mathrm{V}} \pm \Delta_{\mathrm{V}}$	$\theta_{\mathrm{H}} \pm \Delta_{\mathrm{H}}$	$\theta_{\mathrm{F}} \pm \Delta_{\mathrm{F}}$	θ_{rot}	Wall width
1	179±14	88±13	53±30	1.5	[15,28]
2	0±18	90±18	142±33	0.0	[18,29]
3	178±3	88±3	49±5	2.0	[21,37]

Figures 10b and 11b show the result after extracting connected components from the input graphs using the above parameters. With this information, the walls graph has been computed. Notice in Fig. 10b that there is a subgraph which satisfy the conditions stated in definition 5, but does not belong to any textured region. This kind of isolated connected components have been removed and not considered in the wall computation step. See in Figs. 10c and 11c the redrawing of textured regions forming walls using the information contained in the walls graph. It is remarkable in Fig. 11b that, since a narrow range of variation of filling direction has been obtained for this example (see Table 2), a small number of filling edges have been found. However, hatched regions have been properly reconstructed in Fig. 11c.

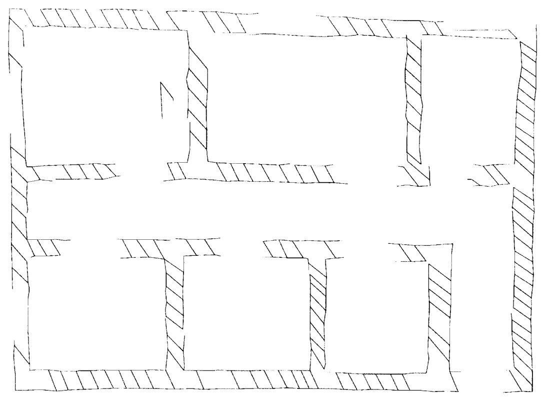
Finally, a matching between the remainder of the input graph and the model graphs has been done. We have worked with a database containing 20 model graphs. Figures 10d and 11d show the result after the recognition and reconstruction of input documents. We can assess that the matching has succeeded despite some accuracy errors such as gaps, splitted points, missing edges, etc. As can be seen in Table 1, the graphs representing each document contain a large set of nodes and edges. This means that graph-matching might take a long time to be computed. If the edges belonging to textured areas are removed, then the graph size is considerably reduced and, thus, the time required by the matching is exponentially reduced. In Table 1 the computation reduction rates are shown. In Fig. 10d we can see another problem that appears after document reconstruction. There are some models which do not fit within the walls because of the distortion of the input hand-drawn document. In this case, it is necessary to warn the user, allowing them to change the size of some elements.

6 Conclusion

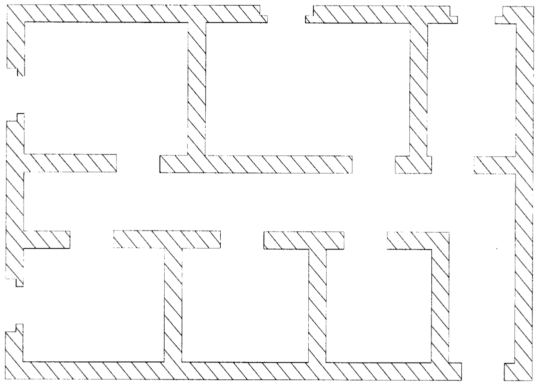
The problem of hand-drawn floor plan document analysis as an alternative CAD input technique has been discussed. An attributed graph-based structure has been proposed to represent the structural information of the document after its vectorization. Document recognition has been stated in terms of a subgraph isomorphism between a graph representing the input document and a set of graphs representing some models for recognition. Computing subgraph isomorphism requires an exponentialy large amount of time. This problem is mainly noticeable when graphs represent disturbed



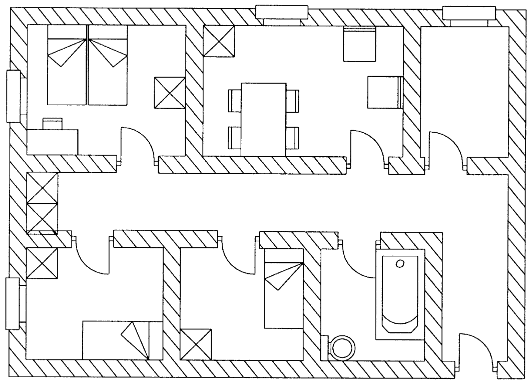
10a



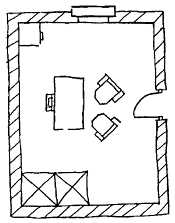
10b



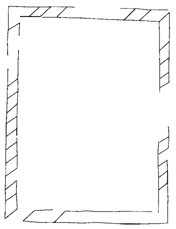
10c



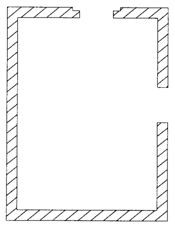
10d



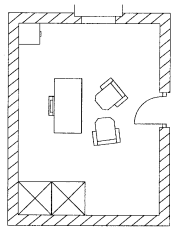
11a



11b



11c



11d

Fig. 10. a Input document; **b** connected components; **c** walls redrawing after parameter computation; **d** document reconstruction after recognition

Fig. 11. a Input document; **b** connected components, **c** walls redrawing after parameter computation; **d** document reconstruction after recognition

line drawings due to hand introduction and the isomorphism must be computed in terms of an inexact graph-matching. To recognize some elements that do not have a fixed pattern, a Hough-transform-based process has been introduced. This process carries out a recognition of walls, characterized by a filling texture based on parallel even-spaced straight lines. This allows not only to recognize the walls, but also to reduce the size of the input graph and, thus, speed up the graph-matching used to recognize other building elements.

Starting from the methods described in this paper, there are two points requiring further research. First of all, to apply the understanding process to drawings of other areas and secondly to generalize the hatching pattern recognition process, in order to detect structured linear textures without a previously defined pattern and being able to redraw them.

References

Antoine D, Collin S, Tombre K (1992) Analysis of technical documents: The REDRAW system. In Baird H.S., Bunke H, Yamamoto K (eds) Structured document image analysis. Springer, Berlin Heidelberg New York, pp 385–402

Boatto L, Consorti V, Del Buono M, Di Zenzo S, Eramo V, Espossito A, Melcarne F, Meucci M, Morelli A, Mosciatti M, Scarci S, Tucci M (1992) An interpretation system for land register maps. Computer 25(7):25–33

Habacha AH (1991) Structural recognition of disturbed symbols using discrete relaxation. In: Proceedings of 1st International Conference on Document Analysis and Retrieval, Saint Malo, France. pp 170–178

Henderson TC (1990) Discrete Relaxation Techniques. Oxford University Press, Oxford

Illingworth J, Kittler J (1988) A survey of the Hough transform. Comput Vision Graphics Image Process 44:87–116

Kasturi R, Bow ST, El-Masri W, Shah J, Gattiker JR, Mokate UB (1990) A system for interpretation of line drawings. IEEE Trans Pattern Anal Mach Intell 12(10):978–992

Kuner P, Ueberreiter B (1988) Pattern recognition by graph matching. Combinatorial versus continuous optimization. Int J Pattern Recogn Artif Intell 2(3):527–542

Leavers VF (1993) Which Hough transform? Comput Vision Graphics Image Process 58(2):250–264

Lee SW, Kim JH, Groen FCA (1990) Translation-, rotation-, and scale-invariant recognition of hand-drawn symbols in schematic diagrams. Int J Pattern Recogn Artif Intell 4(1):1–25

Lladós J, Martí E (1995) Structural recognition of hand-drawn floor plans. In: Proceedings of the 6th Spanish Symposium on Pattern Recognition and Image Analysis, Córdoba, Spain. pp 27–34

Lladós J, Regincós J, Martí E (1993) Interpretación de diseños a mano alzada como técnica de entrada a un sistema CAD en un ámbito de arquitectura. In: Proceedings of the 3rd Congreso Español de Informática Gráfica, Granada, Spain, pp 33–46

López Krahe J, Pousset P (1988) The detection of parallel straight lines with the application of the Hough transform. In: Proceedings of the 9th International Conference on Pattern Recognition, Rome, Italy. pp 939–941

Mohr R and Henderson TC (1986) Arc and path consistency revisited. Artif Intell (28):225–233

Pao D, Li HF, Jayakumar R (1992) Shapes recognition using the Straight Line Hough Transform: Theory and generalization. IEEE Trans Pattern Anal Mach Intell 14(11):1076–1089

Wagner RA, Fischer MJ (1974) The string-to-string correction problem. J ACM 21(1):168–173

Wahl FM (1989) Deriving Features from Hough Space for Object Recognition and Configuration Estimation. In: Simon JC (ed) From Pixels to Features. Elsevier, North-Holland

Josep Lladós received the degree in Computer Science in 1991 from the Universitat Politècnica de Catalunya and the Master in Computer Vision in 1993 from the Universitat Autònoma de Barcelona. Currently he is a Ph.D. student at the Departament d'Informàtica of the Universitat Autònoma de Barcelona and an assistant professor at the same department. His current research field is document analysis and understanding.

Jaime López-Krahe was born in Zamora (Spain) in 1946. He received the B.S., M.S. and Ph.D. degrees in Psychology as well as in Computer Science from Paris VIII's University in 1979 and the Ph.D. from Madrid University in 1982. He joined the Image Department of E.N.S.T. in Paris and has been involved in research there works since 1976. His research interests include pattern recognition, computer vision, computational and discrete geometry, image understanding, Hough transform and technological developments for human disability. Presently he is a professor in Computer Science at Paris University (Paris VIII).

Enric Martí received the degree in Computer Science in 1986 from the Universitat Autònoma de Barcelona and the Ph.D. degree in Computer Science in 1991 from the same university. Presently he is a lecturer at the Departament d'Informàtica of the Universitat Autònoma de Barcelona. His current research field is document image analysis and understanding and line drawing interpretation.

This article was processed by the author using the LaTeX style file *pljour2* from Springer-Verlag.