MVA '92 IAPR Workshop on Machine Vision Applications Dec. 7-9,1992, Tokyo

Structure Recognition of Table-form Documents on the Basis of the Automatic Acquisition of Layout Knowledge

Qin Luo, Toyohide Watanabe and Noboru Sugie
Department of Information Engineering,
School of Engineering, Nagoya University

Furo-cho, Chikusa-ku, Nagoya 464-01, Japan

Abstract

This paper presents a knowledge-based layout structure recognition system. The scope of our system is to identify logical structures from table-form documents without any predefined layout knowledge. The system has two modes: structure acquisition and structure recognition. In the structure acquisition mode, the binary tree, as called the structure description tree, which specifies the logical structure of table-form documents, is created intelligently; and in the structure recognition mode, individual item blocks are extracted and classified successfully by interpreting the structure description tree. In this paper, the structure acquisition method is mainly discussed. In comparison with many traditional approaches based on the preassigned knowledge, our apporach is very applicable and powerful.

1 Introduction

Many researches have reported that knowledge-based document structure recognition methods are very successful to extract and classify the meaningful data from documents automatically. To date, different approaches have been proposed for this work. However, most of these approaches are based on the paradigm that specifies the structure definition data beforehand from external resources, usually through application-specific experts. In order to accomplish document understanding in practice, it is necessary that the structure definition ways should be not only done in simpler and smarter forms but also the layout knowledge should be acquired intelligently by the system itself. In this paper, such a knowledge-based layout structure recognition system is addressed. The scope of our system is to extract the layout structure of table-form documents automatically and then generate a logical structure from the extracted layout structure interpretatively so as to be able to reuse it as layout knowledge.

2 Outline of approach

2.1 Document structure of table-form documents

As shown in Fig.1, table-form documents consist of item fields enclosed with vertical and horizontal line segments. The coordinates of item fields are useful in table-form documents' structure recognition[1]. However, since physical information as coordinates cannot express the adjacent relationship among item fields, its application is rather limitted. For example, Fig.1 (a) and (b) will be considered as the same kind of table-form documents, but they are different in terms of their geometric positions, area sizes and so on. The structure recognition method using physical information can only be applied to those which have the same forms and same structures. When document patterns are put in with some bias angles, some expanded/reduced forms, or some variances of item fields, this method is not adaptable. Document structure knowledge should have sufficient generality and applicability. Our goal of automatically acquiring document structures is to identify the structural characteristics, but not to produce the coordinates of item fields[2],[3].

(a)

(b)

Fig.1 Examples of table-form document.

2.2 Structure description tree[2]

We use the structure description tree to describe the layout structure of table-form documents logically. Structure description tree consists of global structure tree and local structure tree, which define the global and local characteristics of table-form documents, respectively.

(1)Global Structure Tree : A table-form document consists of several blocks, which are meaningful sets of adjacent item fields. The global structure tree describes blocks as structural units: some item field controls more than 1 other item fields logically. The node corresponds to a block. The nodes are divided into 3 node types according to the existence and direction of a repeated structure: vertically repeated node (D); horizontally repeated node (R); and non-repeated node (S). The branches describe the adjacent relationship among blocks. When a table-form document is scanned from upper left corner, the left and right branches link the blocks which are located respectively at the lower and right sides of the current block, as shown in Fig.2(a).

(2)Local Structure Tree: The local structure trees describe the internal structures of blocks. The node indicates that

item fields connected either horizontally or vertically when a block is scanned from the upper left corner. Namely, the nodes are divided into vertical nodes (v), horizontal nodes (h) and terminal nodes (t). When the parent node is a vertical node, the upper and lower item fields for the cutting line are connected to the left and right branches, respectively. When the parent node is a horizontal node, the left and right item fields are connected to the left and right branches, as shown in Fig.2(b). The item field that cannot be further divided is a terminal node.

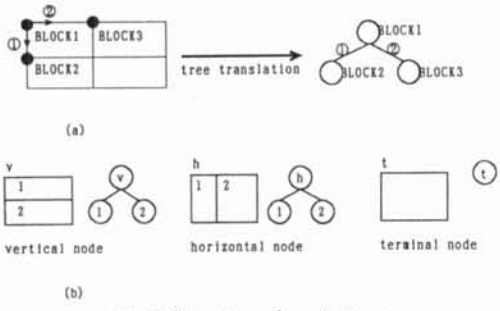

Fig.2 Structure description tree.

2.3 Automatic acquisition of layout knowledge

The acquisition process of layout knowledge is shown in Fig.3. This process is mainly composed of structure decomposition, structure extraction and knowledge composition phases. In the structure decomposition phase, first the horizontal and vertical line segments are extracted and then item fields are identified on the basis of these line segments. Next, item fields containing printed characters are grouped into name fields and other item fields are data fields. In the structure extraction phase, the dependent and meaningful relationships among them are established, using generalized composition rules. The generalized composition rules are a kind of meta-knowledge, applicable to the interpretation of layout structures for many classes of table-form documents. Finally, in the phase of knowledge composition, according to the information obtained from previous phases global structure and local structure trees are produced.

3 Dependent relationship among item fields

There are dependent relationships between name fields and data fields. We define that a data field "a" is dependent on a name field "A" if "A" and "a" have a dependent relationship. The dependent relationship is divided into 4 classes, as shown in Fig.4: single horizontal dependency, single vertical dependency, multiple horizontal dependency and multiple vertical dependency. The relationship is assigned to the mutual item fields by the arrow.

The structure fragment is defined for dependent item fields. Fig.5 shows some examples of structure fragments. (h) and (i) show hierarchical-form types. (j) is an array-form type and (k) shows a structure fragment of single name field. These structure fragments can be explained using 4 dependent relationships. For example, in the array-form type (j), the data fields "d.b" and "e.b" or "d.c" and "e.c" are vertically denpendent on the name field B or C, respectively; and data fields "d.b" and "d.c" or "e.b" and "e.c" are horizontally dependent on the name field D or E, respectively. And also, name fields "B.C" or "D.E" are horizontally or vertically dependent on the name field A. In such way, item fields are constructive elements of structure fragments. The

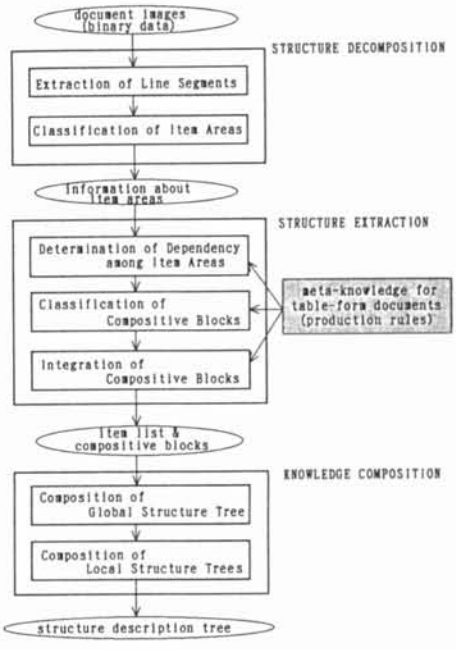

Fig.3 Processing flow.

structure fragments contruct blocks with the meaningful correspondece and adjacent blocks also compose a table-form document as whole.

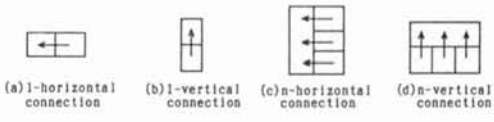

Fig.4 Dependency relationships.

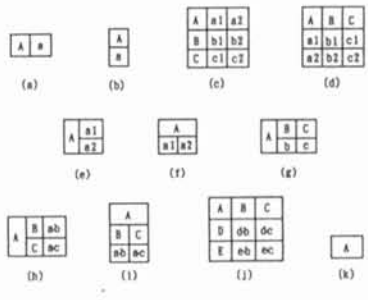

Fig.5 Structure fragments.

4 Structure decomposition

The structure decomposition phase includes 3 procedures: (1) extraction of line segments; (2) identification of item fields; and (3) classification of item fields. We used the boundary distribution recursive algorithm[5] to extract line segments. Areas enclosed by line segments are looked upon as item fields. The information attended to item fields includes mainly a field class, coordinate values of the upper left and lower right corners, etc. These information are useful in order to identify individual document components and generate the structure description tree. Fig.6 shows a distin-

guished result for the table-form document shown in Fig.1 (a). The black and white blocks are identified as name fields and data fields, respectively.

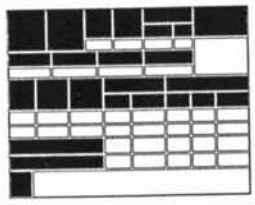

Fig.6 Descrimination of item areas.

5 Structure extraction

The structure extraction phase is such a process which recognizes the structure fragments and blocks according to their depentent relationships. The generalized composition rules are applicable to interpret results in the structure decomposition phase. The generalized composition rules are meta-knowledge about layout structures of table-form documents. We use the production rule to describe them. In order to obtain high processing efficiency, these production rules are controlled in 3 groups.

5.1 Determination of dependent relationship

The following rules are applied to determine the dependent relationships among item fields. The symbols *di* and *dj* indicate data fields, and *tj* represents a name field.

(a-1) IF *di* is located at right side of *tj* directly,
THEN *di* is single-horizontal-dependent to *tj* ;
(a-2) IF *di* is located at right side of *dj* directly,
THEN *di* is single-horizontal-dependent to *dj* .

Similar rules can be used to determine the vertically dependent relationship.

5.2 Classification of structure fragments

Item fields are combined meaningfully as some structure fragments. Because of different dependent relationships, structure fragments are divided into 11 classes, as shown in Fig.5.

(1) Combination of Structure Fragments
We introduce the item list *Lj[tj, dj1, ..., djn]*, which describes a group of linked item fields. They are mutually linked in the horizontal or vertical direction, and data fields *dj1, ...,djn* have the same horizontal (or vertical) dependency. The following rules show the composition method of structure fragments:

(b-1) IF *Lj[tj, dj1, ..., djn]* exists,
THEN item list forms a structure fragment *Fj[Lj]*.
(b-2) IF *Li* and *Fj[Lj1, ..., Ljn]* exist,
IF *Lj1* is directly linked to *Li* from right or lower side,
THEN *Fj[Lj1, ...,Ljn]* becomes *Fj[Li, Lj1, ..., Ljn]*.
IF *Li* is directly linked to *Ljn* from right or lower side,
THEN *Fj[Lj1, ..., Ljn]* becomes *Fj[Lj1, ..., Ljn, Li]*.

(2) Classification of Structure Fragments with Repeating Property
If a structure fragment consists of an item list which has several data fields, it is determined as repeating structure. 5 types (a)-(d) and (j) in Fig.5 are distinguished using the following rules.

(b-3) IF data fields *dix* ($1 <= i <= m$) of *Li[ti, di1, ..., dim]* are single-horizontal-dependent on *ti*,
IF m=1
THEN structure fragments in *Li* belong to type(a).
IF $m > 1$
THEN structure fragments in *Li* belong to type(c).

The rule of data field *dix* being single-vertical-dependent to *ti* can be similarly used to recognize types (b) and (d). The array-form type (j) has a repeating property in two-dimensional directions at once.

(b-4) IF *Fi* and *Fj* belong to types (c) and (d),
and data field of *Fi* is data field of *Fj* at once,
and vice versa,
THEN combine *Fi* and *Fj* to reconstruct a new type of structure fragment of type (j).

(3) Classification of Structure Fragments with Hierarchical Property
By multiple-horizontal-dependency and multiple-vertical-dependency, we can recognize the structure fragments of types (g), (h) and (i).

(b-5) IF *Fi* belongs to type (b) and it's name field is multiple-horizontal-dependent on *tj*,
THEN combine *tj* and *Fi* to reconstruct a new structure fragment of type (g).
(b-6) IF *Fi* belongs to type (c) and it's name field is multiple-horizontal-dependent on *tj*,
THEN combine *tj* and *Fi* to reconstruct a new structure fragment of type (h).

Obviously, type (i) can be recognized using the rule similar to the rule (b-6). Applying the above rules, item fields which have not been combined to some structure fragment are organized as structure fragments of types (e) and (f).

(b-7) IF *di1, ...,dim* are data fields, *dix+1* is directly linked to *dix*($1 < x < m$)from lower side,
and *di1, ...,dim* are multiple-horizontal-dependent on *ti*,
THEN combine it as a new structure fragment *Fi[ti, di1, ...,dim]*with type (e).

A similar rule is useful to recognize type (f). All the item fields which cannot be recognized using above rules are defined as type (k).

5.3 Construction of blocks

Some linked structure fragments have meaningful connection. The following rules transform blocks of types (c) and (h) into a new block:

(c-1)IF fragment exists,
THEN *Fi* can construct a block *Bi[Fi]*.
(c-2)IF *Fi* and *Bj[Fj1, ...]* exist,
Fi and *Fj1* belong to type (c) or (h),
and *Fj1* is directly linked to *Fi* at lower side
THEN *Bj[Fj1,...]* becomes *Bj[Fi, Fj1, ...]* after combining *Fi*.
(c-3)IF *Fi* and *Bj[..., Fjn]* exist,
Fi and *Fjn* belong to type (c) or (h),
and *Fi* is directly linked to *Fjn* at lower side
THEN *Bj[..., Fjn]* becomes *Bj[..., Fjn, Fi]* after combining *Fi*.

Similar rules are useful in the process of combining structure fragments of types (d) and (i) or types (b) and (g).

6 Construction of structure description tree

The characteristics and connective relationships of items fields, structure fragments and blocks, have been already recognized in the structure decomposition and structure extraction phases. They are finally expressed as a structure description tree so as to be applicable to structure recognition of table-form documents. We describe the algorithm of constructing structure description trees briefly.

First, the global structure tree is composed. The node of a global structure tree corresponds to the block, whose branches correspond to the siblings of other blocks. This composition is done recursively. This first step is to find the most upper left corner of block. If it has vertical-repeating property,the node D is assigned.If it has horizontal-repeating property, the node R is so. Otherwise, the node S do so. Next, the right-hand side block for the current one is determined as the right child node, and also the buttom side block does as the left child node. The connected node is repeatedly checked by the above procedure until all blocks are built as nodes. Fig.7(a) shows the global structure tree derived from Fig.1(a), and Fig.7(b) expresses the corresponding global layout structure.

The detail structure of every block is expressed by a local structure tree. The node of the local structure tree corresponds to an item field or a structure fragment, and the branch corresponds to the connective relationship between item fields and structure fragments. When the most left item field or structure fragment has horizontally dependent relationship with the remaining parts, the node "h" is assigned to it. That is, the item field or structure fragment at the left side corresponds to a left child of node "h", and that at the right side corresponds to a right child. Similarly, node "v" is used to link item fields and structure fragments with vertical dependency. This procedure is executed recursively until all terminal nodes reached to item fields. Especially, for repeated structure fragments, only item fields located at the first row or column are recorded in the local structure tree. The indicator and number of repetition will be provided to the corresponding nodes in the global structure tree. Fig.8 shows the local structure tree, corresponding to the block 1 labeled in Fig.7(a).

(a) global structure tree (b) global structure for table-form document in Fig.1(a)

Fig.7 Composition global structure tree.

Fig.8 Composition of local structure trees.

7 Recognition experiment

Our experimental system is implemented on Sun 4/Sparc station+1, using a programming language C. We recognized the table-form document in Fig.9 by means of the structure description tree generated automatically from the table-form document in Fig.1(a). By filling the table-form document in Fig.1(b) with some item data, we get Fig.9. The table-form documents in Fig.1(a) and (b) have the same layout conceptually although there are lots of changes for the field sizes and adjacent relationships among item fields. The result is shown in Fig.10. In Fig.10, the first column contains practical data fields, and the second and third columns are the corresponding name fields which have vertical and horizontal dependency, respectively.

支出官殿	請求者	所属	官職	氏名		命令権者
				番号	印	渡辺
		情報	助手	A217		
概算額	精算額	追給額		返納額		
月日	出発地	到着地	車賃		鉄道賃	
			定額	実費	路程	運賃	その他
10/5	名古屋	東京		3,500		8,500
合計				¥3500		¥8500
請求額				3500		8500
備考						

Fig.9 A used table-form document.

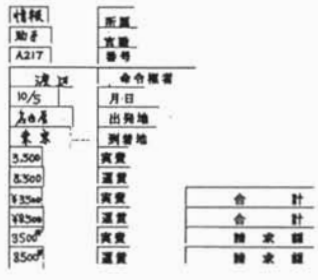

Fig.10 Identified and extracted item areas.

8 Conclusion

This paper presented a structure recognition method of table-form documents on the basis of the automatic acquisition of layout knowledge. Our method is very intelligent because the layout knowledge must not be specified in advance. We recommend to refer to our paper [2] with respect to the structure recognition technique, if necessary.

References

[1] Y.Nakano, H.Fujisawa, O.Kunisaki, K.Okada & T.Hananoi: " Understanding of Tabular Form Documents Cooperating with Character Recognition", EIC trans., Vol.J69-D, No.3, pp.400-409(1986)[in Japanese].

[2] T.Watanabe, H.Naruse, Q.Luo & N.Sugie: "Structure Analysis of Table-form Documents on the Basis of the Recognition of Vertical and Horizontal Line Segments", Proc.of 1st ICDAR, pp.638-646(1991).

[3] T.Watanabe, Q.Luo & T.Fukumura: "A Framework of Layout Recognition for Document Understanding", Proc.of DAIR, pp.77-95(1992).

[4] H.Kojima & T.Akiyama: "Table Recognition for Automated Document Entry System", SPIE, Vol.1384 in "High-speed Inspection Architectures, Barcoding, and Character Recognition", pp.285-292(1990).