International Journal of Document Analysis (2006)
DOI 10.1007/s10032-005-0008-3

ORIGINAL PAPER

Akira Amano · Naoki Asada · Masayuki Mukunoki · Masahito Aoyama

Table form document analysis based on the document structure grammar

Received: 24 July 2004 / Revised: 2 November 2005 / Accepted: 9 November 2005

Abstract Structure analysis of table form documents is an important issue because a printed document and even an electronic document do not provide logical structural information but merely geometrical layout and lexical information. To handle these documents automatically, logical structure information is necessary. In this paper, we first analyze the elements of the form documents from a communication point of view and retrieve the grammatical elements that appear in them. Then, we present a document structure grammar which governs the logical structure of the form documents. Finally, we propose a structure analysis system of the table form documents based on the grammar. By using grammar notation, we can easily modify and keep it consistent, as the rules are relatively simple. Another advantage of using grammar notation is that it can be used for generating documents only from logical structure. In our system, documents are assumed to be composed of a set of boxes and they are classified as seven box types. Then the box relations between the indication box and its associated entry box are analyzed based on the semantic and geometric knowledge defined in the document structure grammar. Experimental results have shown that the system successfully analyzed several kinds of table forms.

Keywords Form processing · Document models · Document analysis systems

1 Introduction

Various kinds of form documents such as research grant application sheets are in circulation around us, and one completes them by filling in the personal data, e.g. name and affiliation, together with the document specific data, e.g. research purpose and budget plan (Figs. 1 and 2). Many people desire to process such a document not on paper but on a computer by using a word processor that promises a high-quality finished product and provides an easy to edit document.

Considering the life cycle of form documents, first, they are generated by their creators, and sometimes they are modified, as the required information may change. After distributing the form, users first read the form, then fill in appropriate information in each field. Finally, they are read or analyzed by others and sometimes stored into databases. Although many forms are produced electronically these days, it is difficult to process them automatically as they do not hold their structural information explicitly. Therefore, it is very important to retrieve structural information from existing forms and define representation of them as well (Fig. 3).

Table form structure analysis has been studied for a long time [1, 2]. For printed documents, detection of rules and extraction of the user filled-in data were conducted [3, 4]. Another type of research is the form identification based on the image features [5, 6]. In other research, identifications are extended to use boxes or cells [7, 8]. As the form identification method using box structure requires structure representation based on the boxes, several structure representations based on the geometrical adjacency of boxes have been proposed [9–12]. Some researches have extended the adjacency analysis to the logical structure analysis, which incorporates a priori knowledge. In most research, knowledge of the form structure is represented by a set of production rules [13–15]. However, as table forms have much variety in their structure, it is difficult to adopt these systems to each document structure as they are constructed on the production rules. To overcome this problem, in the system proposed by Rahgozar et al. [16], graph grammar is used for the representation of the knowledge. In the system proposed by Cracknell et al. [17] or the system proposed by Belaid [18], ordinary grammar is used and even the Unix tool bison is used for the parser in the former one. However, document structure considered in these systems is quite simple compared to prior ones such as [13].

A. Amano (✉)
Graduate School of Informatics, Kyoto University, Yoshida-honmachi, Sakyo-ku, Kyoto, Japan
E-mail: amano@i.kyoto-u.ac.jp

N. Asada · M. Mukunoki · M. Aoyama
Department of Information Sciences, Hiroshima City University, 3-4-1 Ozukahigashi, Asaminami, Hiroshima, Japan

基盤A·B·C(一般)−1

平成14年度　基盤研究(A・B・C)研究計画調書　(新規)

注1. 別途平成14年度基盤研究(A・B・C)(一般)研究計画書作成・記入要領(黄色)を参照して下さい。
注2. ※印の欄は研究機関において記入してください。

※機関番号	
※整理番号	

基盤研究	A　B　C	研究	(1)　(2)	審査区分	

審査希望部門	部	分科	細目	部・分科細目番号	広領域	系	分割番号
						人・物 化・生	1 2

基盤研究(C)での申請のうち、部・分科・細目番号「632」「635」「636」「652」「658」を選択した場合、分割番号のどちらか一方を必ず○で囲んでください。

研究代表者氏名	印	所属研究機関・部局・職	

研究課題	

研究経費 千円未満の端数は切り捨てる	年度	研究経費(千円)	使用内訳(千円) 設備備品費	消耗品費	旅費	謝金	その他
	平成14年度						
	平成15年度						
	平成16年度						
	平成17年度						
	総計						

Fig. 1 An example of table form document which is a part of a grant application sheet by Japanese government

NAME		POSITION TITLE	
TITLE OF PROJECT			
	TOTAL	ITEM	
		EQUIPMENT	TRAVEL
YEAR 1st			
YEAR 2nd			
TOTAL			

Fig. 2 Another example of table form document

For the logical representation of documents, the most famous representation is SGML [19] and XML. Both formats define attributes of words or sentences and also structure of documents. However, as both formats provide only the description of a basic structure, it is difficult to represent complex form structures by using only these. As with other XML-based formats, we need additional formats to describe complex forms. Other industrial representations such as Document Management Alliance (DMA) [20] and Open Document Architecture (ODA) [21] also scope on document handling but do not offer the capability of complex structural representation. In the research by Cracknell et al. [17], a representation format of the logical structure for the form was proposed. The aim of this format is quite similar to ours; however, the format has no consideration for the two-dimensional (2D) structure of the table part of the forms.

To cope with these problems, in this paper, we propose a method to retrieve structural information from existing documents based on document structure grammar and then propose a representation of table form documents TFML.

As the forms are tools for informing what and how information is to be filled-in on paper, it can be recognized as a language. Therefore, we can use the underlying grammar for recognizing the structure of form documents. The target form of this paper is table form documents. They are decomposed into a set of fields, and those relations are analyzed based on the semantic and geometric knowledge defined in the document structure grammar. One advantage of using a grammar-based approach rather than a production system is that a number of if-then descriptions like [13–15] is difficult to keep consistent when the system is reconfigured. Another advantage is that it can be used for document generation, modification, and even for correction.

In the following section, we first analyze the table form documents from the communication language point of view and then present the document structure grammar. We propose a table form document analysis system based on the grammar and then present an XML-based representation of table forms. Finally, experimental results of analysis are presented.

2 Document communication model and table form document

Documents have their own function, and from this point of view, they can be classified into two categories: *Informational Documents* and *Interactive Documents*. The Informational Documents are the ones that deliver some fixed information to the readers, for example newspapers

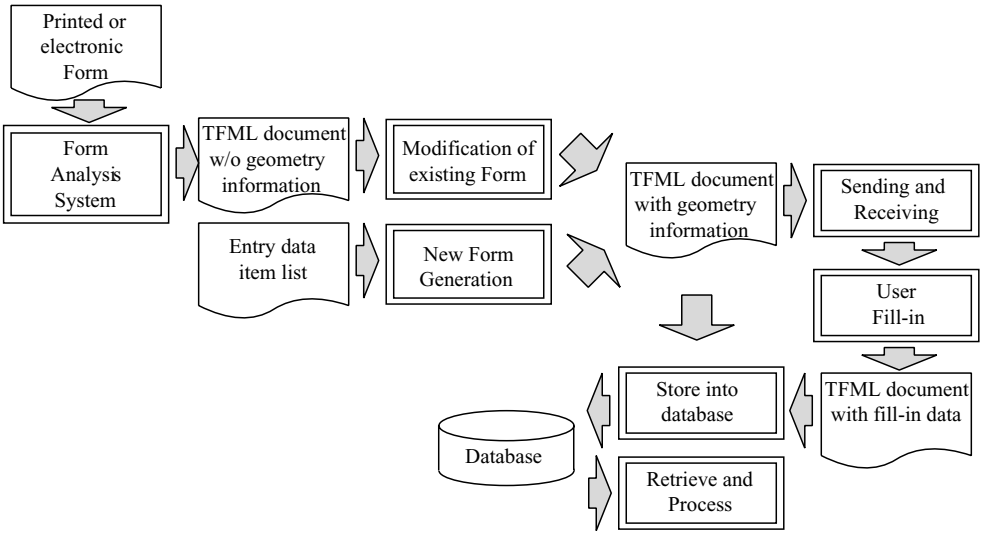


Fig. 3 Form processing and TFML

or commercial leaflets. From the communication point of view, these documents have complete information, and the communication direction is one way. Therefore, we can call them complete documents. On the other hand, the Interactive Documents are the forms on which we need to fill in appropriate information and send them to others for further processing.

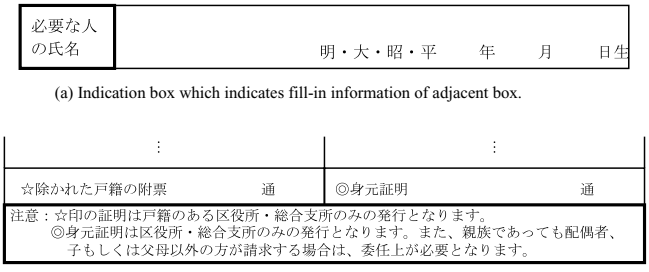
The target document of this paper is Interactive Documents. As the Interactive Documents can be considered as some sort of sentence in a communication language using forms, we may be able to determine the underlying grammar of them. In this paper, we propose a grammatical description of the Interactive Documents and an analysis method for them. One important aspect of this paper is to reveal the grammar which we are using when we communicate by such forms. By establishing such grammar, we can use it for analyzing, modifying, reformatting, error correcting and in many other applications.

2.1 Communication functions of each element in the interactive documents

The Interactive Documents consist of communication elements and their structures. Here we classify communication elements into three categories by their function.

2.1.1 Indication and explanation function

In the interactive documents, document creators prompt readers to fill in appropriate information at certain places. We call the function of elements which indicate the type, place, or format of information to fill in "Indication and Explanation function". In Fig. 4a, an element with thick lines indicates the fill-in information of the adjacent element. In Fig. 4b, the element with thick lines indicates general explanation of the form.



(a) Indication box which indicates fill-in information of adjacent box.

(b) Explnation box which has general description of the document.

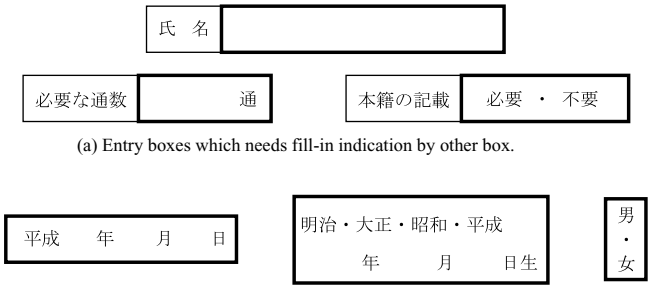
Fig. 4 Elements which have indication and explanation function

2.1.2 Entry function

We call the function of the elements which indicates the fill-in position "Entry function". There are four sub-categories for this function: (1) Complete fill-in: readers must write whole information (2) String insertion: readers must insert strings between the preprinted strings (3) Selection: readers must select one or more strings and write, circle or mark them (4) Paste: readers must paste pictures or images to the element. Two types of elements which have entry function are shown in Fig. 5. Note that the elements in Fig. 5a require fill-in indication from other elements, in this case elements with thin line, and the elements in Fig. 5b do not need indications.

2.1.3 Example function

In some case, there exists ambiguity in fill-in formats. For example, format of a date can be written in Christian era or domestic era. When the form creator wants to specify these instructions, sometimes they prepare preprinted strings for an example. Here we call this kind of instructing function "Example function".



(a) Entry boxes which needs fill-in indication by other box.

(b) Entry boxes which do not need fill-in indication by other box.

Fig. 5 Elements which have entry function

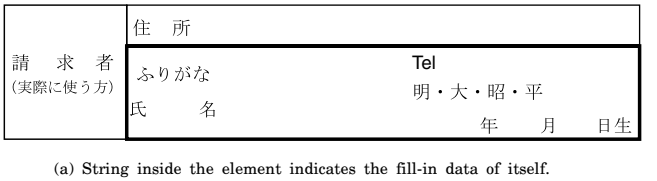
年 度	諸 経 費
平成 １３ 年度	円
年度	円
年度	円

(a) Element which has a fill-in string as an example.

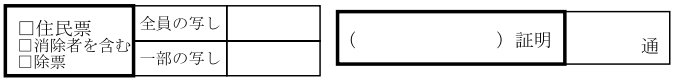
受講場所	広島市安佐南北区倉掛二丁目３３番１号 広島総合防災センター

(b) Element which has information to the writer.

Fig. 6 Elements which have example function

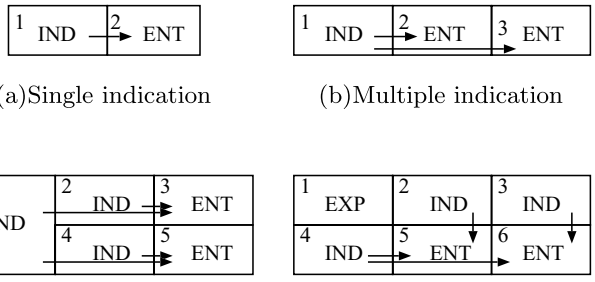


(a) String inside the element indicates the fill-in data of itself.



(b) By filling-in the data to the element, it indicates the fill-in data of right adjacent element.

Fig. 7 Elements which have indication and explanation function and fill-in function



(a)Single indication (b)Multiple indication

(c)Hierarchical indication (d)Bidirectional indication

Fig. 8 Four indication structures

In Fig. 6a, elements with thick lines indicate the fill-in format of the following elements. In Fig. 6b, elements with thick lines provide information to the writers.

In real-form documents, many elements do not only have one function but also two or more functions described earlier. For example, the elements shown with thick lines in Fig. 7a have several strings inside which indicates the fill-in data. In Fig. 7b, the strings inside the elements have a fill-in function. By filling-in the check mark or number, it indicates the fill-in data of the adjacent element.

Note that this classification of element function in the form document is not only restricted to the table forms, but also to general form documents.

2.2 Structure of table form documents

The forms can be classified into two types: one is free format form documents and the other is table form documents. The characteristic of the table form documents is that every element in the document is a rectangular field formed by horizontal and vertical rules. In this paper, we call these elements “boxes”. Elements in both documents have one or more functions as described in the previous section.

In the table form documents, fill-in information to a box is indicated by one or more boxes which has the indication and explanation function. We call this structure “indication structure”.

There is a good classification of the adjacent indication structure in [13]. Here, we reclassify them into recursive combinations with the following four structures.

1. *Single indication* (Fig. 8a). One box which has entry function is indicated by one box which has indication function.
2. *Multiple indication* (Fig. 8b). Multiple entry boxes are indicated by one indication box.
3. *Hierarchical indication* (Fig. 8c). One entry box is indicated by two or more indication boxes which are hierarchically structured. This structure can be understood as a recursive structure of the single indication.
4. *Bidirectional indication* (Fig. 8d). One entry box is indicated by two independent indication boxes which are placed horizontally and vertically.

In this paper, we deal with these four types of indication structures.

2.3 Box type

Every box has at least one function described in Sect. 2.1. From the analysis of 81 public table form documents, we were able to classify them into seven types which have a single function or combinations of two functions. We call them *box types* which are described as follows:

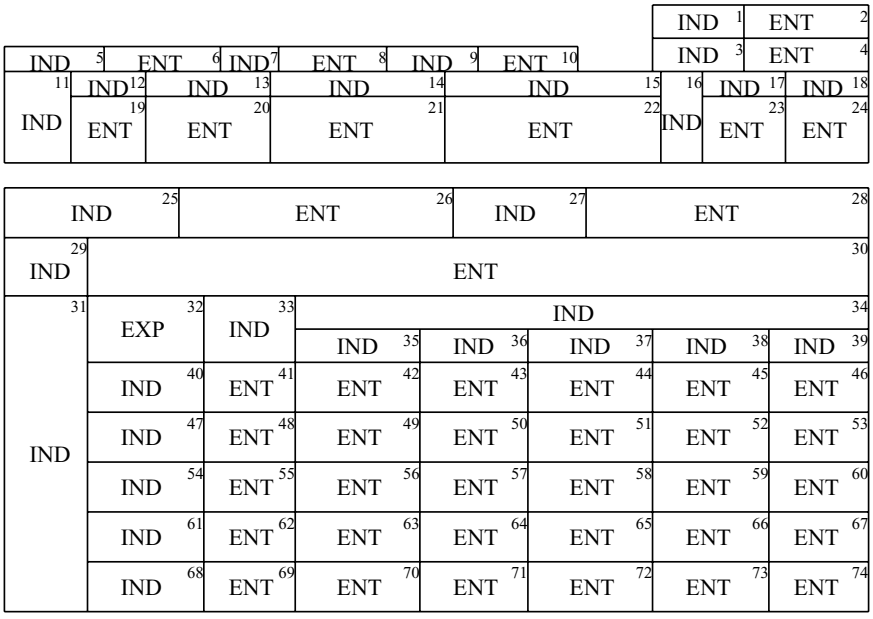


Fig. 9 Box types of Fig. 1

Indication box (IND). This box indicates the fill-in information of other boxes and the box needs no fill-in indication.

Explanation box (EXP). This box has general explanations on the document in it. Therefore, it needs no fill-in information and needs no indication from other boxes.

Entry box (ENT). This box is a fill-in box which needs fill-in indication from another box, and also needs fill-in information from its nature. However, it does not have indication function.

Example box (EXM). This box has example fill-in strings in it. Therefore, it needs indication information from other boxes, but needs no fill-in information and does not have indication function.

Indication entry box (IEN). By filling in some information, this box indicates fill-in information of other boxes.

Self-indication entry box (SIE). This box is an entry box which has indication information written in. Thus it needs no fill-in indication from other boxes, but needs fill-in information.

Need no entry box (NNE). This box is some kind of blank box which needs no fill-in indication information from other boxes, and needs no fill-in information.

For example, box types of each box in Fig. 1 becomes Fig. 9, and box types of Fig. 2 becomes Fig. 10.

From the structure analysis point of view, we can classify these box types by the aspect that the box will indicate fill-in information of other boxes and that the box needs indication from other boxes. The resulting classification into four categories becomes as Table 1.

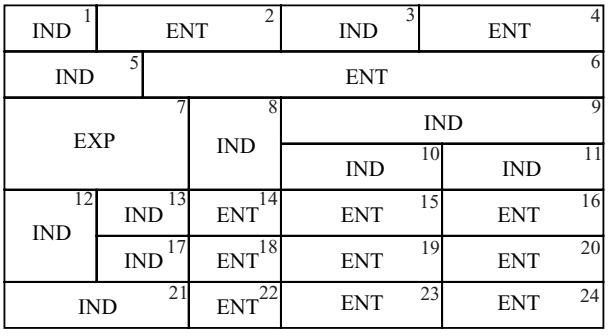


Fig. 10 Box types of Fig. 2

2.4 Unification of adjacent boxes into a compound box

If two adjacent boxes have the same height or same width, we define that they are "unifiable". In certain situations, unifiable boxes can be combined into one box and treated as a single box. We call this unified box a "compound box". By unifying two adjacent boxes recursively, we can analyze the structure of table form documents. Note that there exists horizontal and vertical (Fig. 11a, b) unifications.

Table 1 Classification of box type in the table form documents

Has indication function	No indication function
Need indication	
–	Entry box (ENT)
	Example box (EXM)
No indication is necessary	
Indication box(IND)	Self-indication entry box (SIE)
Indication entry box(IEN)	Explanation box (EXP)
	Need no entry box (NNE)

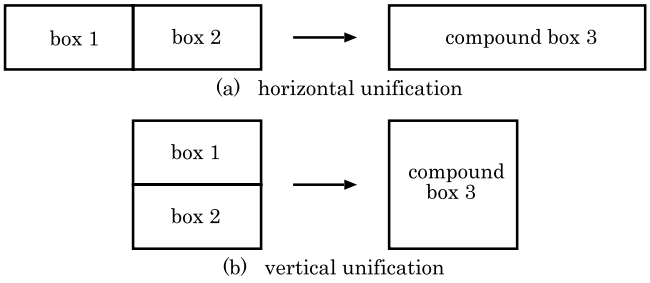


Fig. 11 Two unifiable boxes and resulting compound box

By the combination of two unifiable box types, we define two types of resulting compound boxes.

1. Indication compound box (icb)
   As in the case of a compound box with an indication box and an entry box, this compound box has an indication function.
2. General compound box (gcb)
   When two boxes which have no indication functions are unified, the resulting box has no indication function.

3 Document structure analysis

3.1 Overview of the table form document analysis

As described in the previous section, we assume that a form is composed of multiple boxes. The goal of this table form document structure analysis is to obtain relations between two boxes which have a logical relation.

For the printed documents, the analysis becomes as follows.

1. Box detection
2. Box type classification
3. Grammar-based document structure analysis

Box detection is a process which detects all the rectangular fields in the document by detecting horizontal and vertical rules. We first used the simple Hough transform method to detect the longest vertical line in the document, then corrected the rotation of the scanned document by using its angle. Then we used projection onto the x and y axis to detect vertical and horizontal rule positions. By detecting the horizontal and vertical lines at the detected positions, we detected rule lines forming each box.

Box type classification is a process which determines the box type of each detected box. It is very difficult to obtain complete results of box type classification. Therefore we are currently using a semi-automatic classification tool for this purpose. We used OCR software to determine the words preprinted in each box. By providing a simple database, box types are determined by matching the preprinted words with the database. After this process, the result is shown to the user, and the user corrects the box types if it is misclassified.

Finally, logical structure is analyzed with a grammar-based document structure analysis algorithm which is described in the following section.

Note that many form documents are provided in electronic files such as PDF files. For these documents, box detection and OCR process is not necessary. However, the box type classification process is still necessary and further research should be done on this process.

3.2 Two-dimensional data analysis with one-dimensional grammar

As the distribution of boxes in a table form document is two dimensional, data structure of a document naturally becomes 2D. Therefore, we can simply use Graph Grammar Parser to analyze each document with the grammar by extending proposed grammar to the graph grammar. However, such graph grammar notation becomes very large because each rule of the grammar will be a combination of our one-dimensional (1D) production rules. On the other hand, except for the bidirectional indication structure, unification of adjacent boxes can be written as 1D rules of a grammar. Bidirectional indication structure can be recognized as a combination of two multiple indication structures which have different indication directions. Therefore, we can analyze them by applying two different directional grammatical analyses. By using 1D rules, we can easily understand the whole structure of the grammar as it is much simple than graph grammar notation.

3.3 Document structure grammar

We consider three indication structures; single, multiple and hierarchical. We denote the starting symbols as `<document>`. Seven terminal symbols are denoted as `IND, IEN, EXP, ENT, EXM, SIE, NNE` which corresponds to the box type in Table 1. We denote two unifiable boxes by adding symbol "•" between them. We used the following three symbols as nonterminal symbols.

`<document>` is the start symbol that represents the whole document.
`<icb>` denotes the indication compound box that implies the indication to adjacent entry box.
`<gcb>` represents the general compound box that has no indication to other boxes.

Figure 12 shows the document structure grammar described by the extended BNF notations where "::=", "|" and "ϕ" denote derivation, selection and null element, respectively.

Single indication structure can be denoted as a unification rule of two adjacent boxes which have both indication function and fill-in function. Multiple indication structure can be denoted as a recursive structure of unification of adjacent boxes which have both indication function and fill-in function (Fig. 13). Hierarchical indication structure can

1 : <document> ::= ϕ | <document> <gcb>

2 : <gcb> ::=

3 : IND | IEN | SIE | EXP | NNE | <icb>

4 : | <gcb> • <gcb>

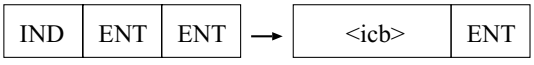
5 : <icb> ::=

6 : IND • (ENT | EXM | <gcb>)

7 : | IEN • (ENT | EXM | <gcb>)

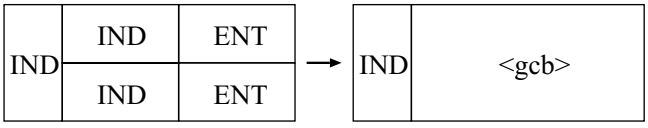
8 : | <icb> • (ENT | EXM | <gcb>)

Fig. 12 Document structure grammar



<icb>::=IND • ENT | <icb> • ENT

Fig. 13 Single and multiple indication structure



<gcb>::=<icb> | <gcb> • <gcb>
<icb>::=IND • ENT | IND • <gcb>

Fig. 14 Hierarchical indication structure

be denoted as a unification rule of one indication box and one general compound box which have two or more general compound boxes inside (Fig. 14).

For the bidirectional indication structures, we can analyze the part under the second row and the part in the right of the second column as multiple indication structures. For analysis of first row and column, we prepare rules which reduce multiple adjacent indication boxes into one <gcb>.

With this grammar, we can analyze table form documents which are constructed of four indication structures. Note that in the analysis process, the priority of reduction to <icb> is set higher than that of <gcb>.

3.4 Horizontal prior analysis and vertical prior analysis

The table form documents have 2D structure, while the grammar in Fig. 12 describes the 1D box relation, hence the production rules should be applied horizontally and vertically. For this purpose, we prepare two kinds of box lists named Hlist and Vlist that represent the horizontally and vertically sorted box queues respectively. The Hlist (Vlist) is made by the following procedure.

1. Sort all boxes in the ascending order of $y(x)$ coordinates then in that of $x(y)$ ones.
2. Insert "•" between the pair of horizontally (vertically) adjacent boxes that have the same height (width).

In the syntax analysis, the reduction rules in the grammar are applied twice on both box lists by switching the priority between Hlist and Vlist, because both results of the horizontal and vertical reductions are necessary to analyze correctly the bidirectional indications in table structure. Note that the reduction of Vlist is accompanied by that of Hlist simultaneously and vice versa to maintain the consistency of box adjacency. We call the parsing, which has higher priority in horizontal unification, "Horizontal Prior Analysis" and vertical unification "Vertical Prior Analysis".

The procedure for Horizontal Prior Analysis is as follows.

1. Select the last rule in Fig. 12.
2. Search the box pattern in Hlist that matches the selected rule. If no pattern is found in Hlist, search for it in Vlist. If no pattern is found anywhere, go to Step 4.
3. Replace the box pattern found in both Hlist and Vlist with a compound box that has a new identifier. If the box pattern consists of two boxes, replace the preceding box and delete the following one in both lists. If the new compound box has the connectivity to adjacent boxes in the lists, insert the adjacency symbol "•" between them. Go to Step 2.
4. Select the next rule in the preceding line in Fig. 12 and go to Step 2. If there is no rule left, then end.

With the above algorithm we can obtain Horizontal Prior Analysis result. By changing the priority of Hlist and Vlist in Step 2, we can obtain Vertical Prior Analysis result.

Result of Horizontal Prior Analysis of 2 by 3 bidirectional structure is shown in Fig. 15. In this example, first, rule <icb>::=IND • ENT, which has highest priority, is applied to box 4 and 5, followed by applying rule <icb>::=<icb> • ENT. Then, rule <gcb>::= EXP, <gcb>::= IND and <gcb>::= <gcb> • <gcb> is applied to box 1 and 2, and box 3 is also unified to them. At this step, there is no applicable rule for Hlist, thus we try to apply rules to Vlist. <icb> is reduced to <gcb>, and then vertically adjacent <gcb>s are reduced to one <gcb> and the analysis ends.

3.5 Indication relation analysis

In this section, for an example of using the result of structure analysis, we describe how to analyze indication relations.

In the parse tree, there is always an indicating box in the left of a node and the box which is indicated by that box is placed in the right. Therefore, we can search all the

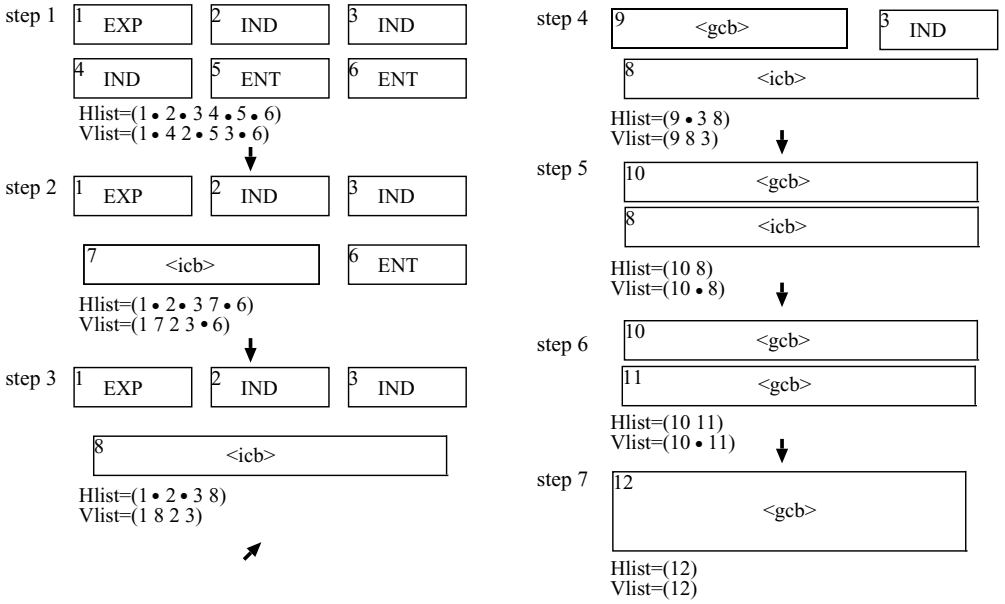


Fig. 15 Example of horizontal prior analysis

indicating boxes by searching the parse tree up and left for each entry box.

Let Box list be the list of <gcb> elements which construct the whole document. The indication analysis algorithm is described as follows.

1. Get the first element from Box list and set to root.
2. Make key list empty.
3. Call index(root).
4. Delete root from Box list. If Box list is empty, algorithm end, else go to Step 1.

The algorithm of function index(root) is as follows.

1. If the box type of root is IND, return this box.
2. If the box type of root is ENT, print key list as indication box list of this box.
3. If the box type of root is SIE, print key list and itself as indication box list of this box.
4. If the box type of root is icb, add return value of index(left child of root) to the last of key list. Call index(right child of root) and delete the last element of key list.
5. If the box type of root is gcb, call index(left child of root) then call index(right child of root) if it exists. With this algorithm, we can obtain all the indication boxes which indicates every entry box.

3.6 Analysis of bidirectional indication with 1D grammatical analyzer

Each entry box in the bidirectional indication structure is indicated by both horizontal and vertical indication boxes. These indication structures can be analyzed individually by Horizontal and Vertical Prior Analysis as shown in Sect. 3.4.

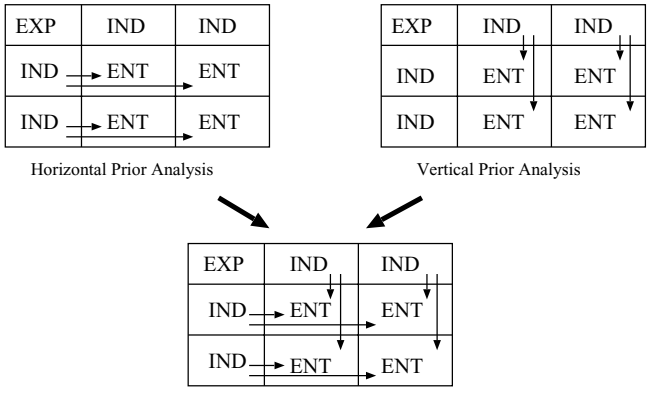


Fig. 16 Combine horizontal and vertical prior analysis results

Indication boxes which indicates each box in the bidirectional structure can be generated by combining results of both analysis (Fig. 16).

Indication box lists obtained for each entry box in Fig. 15 by Horizontal and Vertical Prior Analysis are shown in Table 2. By combining these results, we can obtain all the indication box lists for each entry box in the bidirectional structure as shown in Table 2. Entry box 5 is indicated by indication boxes 2 and 4, and entry box 6 is indicated by indication boxes 3 and 4.

Table 2 Result of box indication analysis of the document shown in Fig. 16

Entry box No.	5	6
Indication box list by Vertical Prior Analysis	2	3
Indication box list by Horizontal Prior Analysis	4	4
Result of combination	2, 4	3, 4

4 TFML: representation of table form

To support handling of table form documents in the process of generation, modification, fill in, and in many other processes, we need general representation of table form documents. Here, we propose XML-based representation of a table form which can handle geometrical information, indication structures, and also the meta-structure of table forms. Meta-structure means logical relation between non-adjacent boxes. One example of meta-structure is that the fill-in data of box A should be the summation of box B and box C where these boxes are not adjacent to each other.

DTD of TFML is shown in Fig. 17. A sample TFML file is shown in Fig. 18 which is generated from the analysis result of the proposed method. The basic structure of the file reflects the indication pattern of the document where geometrical information is embedded as optional information. Meta-information, such as arithmetical relations are also embedded as optional information in each box.

As TFML's representation is general, it can be used as the resulting format of document structure analysis. Furthermore, it can be used as the input definition of a table form generation system, and entry data format of database systems.

5 Experiments

5.1 Experimental results on real documents

The proposed method is applied to the document shown in Fig. 2. As described in Sect. 3.1, we first applied the Hough transformation method to detect document rotation and correcting the image. Then, we detect each rule by detecting x and y position by making projection onto the x and y axis. Finally, by tracing the lines, we detect horizontal and vertical rules and detect every box formed by these rules.

After detecting each box, we applied OCR to detect preprinted words if the inner area of the box is not empty. By matching these words with the box classification database, we get box type candidates of each box. The box type of the box whose inner area is empty is determined as the entry box. Finally, these results are shown to the user, and the user corrects the box type if it is misclassified.

After obtaining the correct box type, our structure analysis is applied. Part of the parse tree generated by the Horizontal Prior Analysis is shown in Fig. 19. The result includes the analysis results of boxes 12–20 which are placed at the lower part of the original document. In the parse tree, multiple indication structure composed by boxes 13, 14, 15 and 16, hierarchical indication structure composed by boxes 12 and 13–16 are analyzed properly. By the Horizontal Prior Analysis, indication boxes 12 and 13 were correctly analyzed to indicate entry box 14, 15 and 16. By the Vertical Prior Analysis, indication boxes 9 and 10 were also analyzed correctly to indicate entry boxes 15 and 19.

```
<!ELEMENT document ((single | multiple |
          hierarchical | table | SIE | EXP | NNE)+)>
<!ELEMENT single ((IND | IEN) , (ENT | EXM))>
<!ELEMENT multiple ((IND | IEN), (ENT | EXM)+)>
<!ELEMENT hierarchical ((IND | IEN)
          ,(single | multiple | hierarchical)
          ,(single | multiple | hierarchical)+)>
<!ELEMENT table ((EXP | ENT)
          ,col_indication , row_indication , entry)>
<!ELEMENT col_indication (indication+)>
<!ELEMENT row_indication (indication+)>
<!ELEMENT indication ((IND | IEN)+)>
<!ELEMENT entry (row+)>
<!ELEMENT row (col+)>
<!ELEMENT col (ENT | EXM)>
<!ELEMENT ENT EMPTY>
<!ELEMENT EXM (#PCDATA)>
<!ELEMENT IND (#PCDATA)>
<!ELEMENT IEN (#PCDATA)>
<!ELEMENT EXP (#PCDATA)>
<!ELEMENT SIE (#PCDATA)>
<!ELEMENT NNE (#PCDATA)>
<!ATTLIST ENT box_num CDATA #IMPLIED
          str_width CDATA #IMPLIED
          str_height CDATA #IMPLIED
          position CDATA #IMPLIED>
          relation CDATA #IMPLIED>
<!ATTLIST EXM box_num CDATA #IMPLIED
          str_width CDATA #IMPLIED
          str_height CDATA #IMPLIED
          position CDATA #IMPLIED>
          relation CDATA #IMPLIED>
<!ATTLIST IND box_num CDATA #IMPLIED
          position CDATA #IMPLIED>
<!ATTLIST IEN box_num CDATA #IMPLIED
          str_width CDATA #IMPLIED
          str_height CDATA #IMPLIED
          position CDATA #IMPLIED>
<!ATTLIST EXP box_num CDATA #IMPLIED
          position CDATA #IMPLIED>
<!ATTLIST SIE box_num CDATA #IMPLIED
          str_width CDATA #IMPLIED
          str_height CDATA #IMPLIED
          position CDATA #IMPLIED>
<!ATTLIST NNE box_num CDATA #IMPLIED
          position CDATA #IMPLIED>
```

Fig. 17 DTD of TFML

The method was applied to 165 sheets in the 81 documents which were provided by the public office in Japan. By assigning the box type to each box in the documents semi-automatically, we successfully analyzed 154 sheets which is 93% of the documents. The main reason for the analysis failure is the shape of the element. The shape of some elements in the table form document are not rectangle but hexagon or a much more complex shape. As such shapes cannot be handled by our system, these documents could

```
<?xml version="1.0" encoding="EUC-JP" standalone="no"?>
<!DOCUTYPE document
          SYSTEM "file:document.dtd">
<document>
  <single>
    <IND box_num = "1">NAME</IND>
    <ENT box_num = "2"/>
  <\single>
  ⋮
  <table>
    <EXP box_num = "7"></EXP>
    <col_indication>
      <indication>
        <IND box_num = "8">TOTAL</IND>
      </indication>
      <indication>
        <IND box_num = "9">ITEM</IND>
        <IND box_num = "10">EQUIPMENT</IND>
      </indication>
      <indication>
        <IND box_num = "9">ITEM</IND>
        <IND box_num = "11">TRAVEL</IND>
      </indication>
    </col_indication>
    <row_indication>
      <indication>
        <IND box_num = "12">YEAR</IND>
        <IND box_num = "13">1st</IND>
      </indication>
      <indication>
        <IND box_num = "12">YEAR</IND>
        <IND box_num = "17">2nd</IND>
      </indication>
      <indication>
        <IND box_num = "21">TOTAL</IND>
      </indication>
    </row_indication>
    <entry>
      <row>
        <col><ENT box_num = "14"/></col>
        <col><ENT box_num = "15"/></col>
        <col><ENT box_num = "16"/></col>
      </row>
  ⋮
```

Fig. 18 TFML representation of Fig. 2 (part)

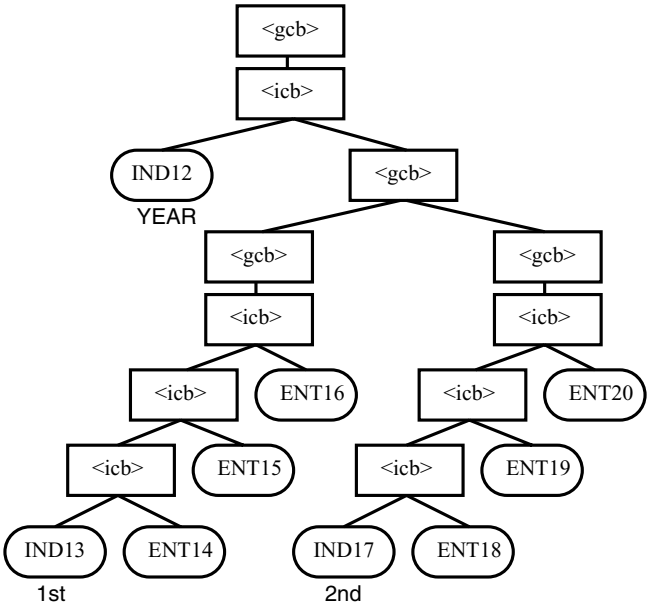


Fig. 19 Resulting parse tree of Horizontal Prior Analysis of Fig. 2 (part)

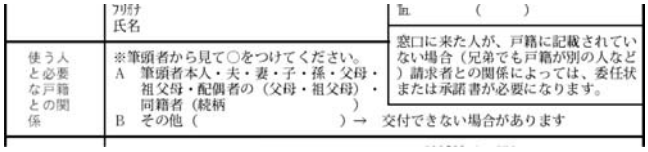


Fig. 20 Failed document 1. The shape of the element in the center do not form rectangle which could not be analyzed with our system

not be analyzed. The other reason for the analysis failure is the indication structure. Two documents had a hierarchical structure which did not form a rectangular area (Figs. 20 and 21). As the indication box of the hierarchical structure can only indicate the rectangular compound box, such structures can not be analyzed with our system.

There was one pattern which was successfully parsed, but the result had ambiguity and at least one of the results can be considered not correct. In the case of Fig. 22a, one of the results of our structure analysis system tells that the left most indication box has indication relation with right most boxes although the left part and the right part have no relation in this case. In another case in Fig. 22b, one of the results tells that the left most indication box has indication relation with two adjacent boxes at the right most position. This happens because the indication compound box (icb) composed of boxes at the left side of the document is related to the general compound box (gcb) composed of boxes at the right side of the document. Actually, in this case, these two compound boxes do not have indication relations. Syntactically, these results can not be omitted without semantic information.

The result of the structure analysis can be used in many applications. Here we made a simple application which generates a synthesized table form document which is a filled-in document of an existing table form document. Each filled-in contents are prepared in the database which is tuples of an indication string and filled-in strings. Sample result of a synthesized document for the document shown in Fig. 1 are shown in Fig. 23.

5.2 Discussion

We have proposed a grammar-based description of the table form documents which can be used in the table form document analysis system.

Generally, documents which can be analyzed by a document analysis system based on a grammar representation, and those which can be analyzed by a system based on the production system are identical. This is because a parser of grammar can be realized by a production system. However, the grammar-based system has the extremely important advantage that the knowledge of the document structure is separated from the analysis system. This means that we can easily modify the knowledge of the document structure implemented in the system, and we even can use this

Fig. 21 Failed document 2. The left element is an indication box which forms a hierarchical structure; however, the right part of the structure does not form a rectangular shape

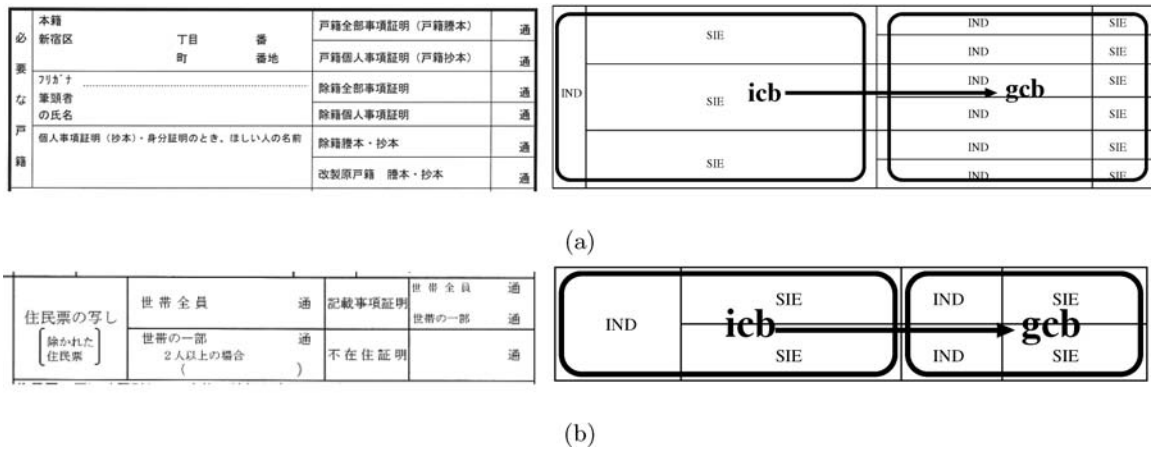


Fig. 22 Left panel shows a document whose structure was successfully analyzed but the result had error. Resulting relations are shown in the right panel. One of the result tells that the left most indication box has relation with 12 boxes in the right part of the document (**a**). Also the left most indication box has relation with 4 boxes in the right part of the document (**b**). This happens because the left part of the document forms an indication compound box and the system recognizes that this icb indicates general compound box in the right part

基盤A·B·C(一般)−1

平成14年度 基盤研究(A・B・C)研究計画調書 (新規)

注1. 別途平成14年度基盤研究(A・B・C)(一般)研究計画書作成・記入要領(鶯色)を参照して下さい。
注2. ※印の欄は研究機関において記入してください。

※機関番号	
※整理番号	

基盤研究	A B (C)	研究	(1) (2)	審査区分	一般

審査希望部門	部	分科	細目	部・分科細目番号	広領域	系	分割番号
	複合領域	情報科学	情報システム学	733		人・物 化・生	1 2

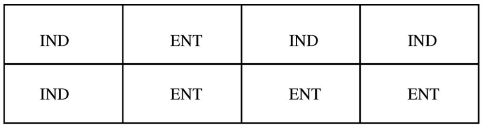
基盤研究(C)での申請のうち、部・分科・細目番号「632」「635」「636」「652」「658」を選択した場合、分割番号のどちらか一方を必ず○で囲んでください。

研究代表者氏名(ふりがな)	天野 晃 印	所属研究機関・部局・職	広島市立大学・情報科学部・助教授

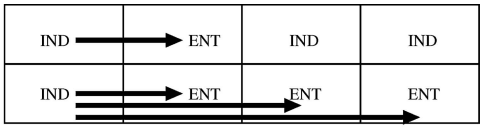
研究課題	書式構造文法を用いた罫線文書の構造解析

研究経費 千円未満の端数は切り捨てる	年度	研究経費(千円)	使用内訳(千円) 設備備品費	消耗品費	旅費	謝金	その他
	平成14年度	2,400	1,500	300	500	50	50
	平成15年度	1,600	700	300	500	50	50
	平成16年度	900	0	300	500	50	50
	平成17年度						
	総計	4,900	2,200	900	1,500	150	150

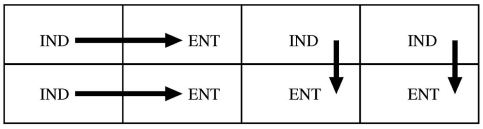
Fig. 23 Sample synthesized document for the document Fig. 1



(a) Result of box classification for this pattern.



(b) One intermediate parse result which will be rejected because two IND boxes in the upper right part do not have relating ENT boxes.



(c) Correct parse result where all IND boxes have their related ENT boxes.

Fig. 24 Example form document pattern which can be easily analyzed by a grammar based system, but needs search control with a production rule based system

knowledge for different purposes, such as form modification system, form correction system, or form generation system.

One example of the above discussion is shown in Fig. 24. In some cases, both a production rule based system and a grammar-based system produce incorrect results such as Fig. 24b shows an intermediate analysis result in the analysis process. Both systems should reject this result and search for the correct one such as Fig. 24c. However, in the production rule based system, we need to explicitly provide the control sequence of this search. In contrast with this, in the grammar based system, we only need to provide document structure knowledge.

Compared with other grammar based document analysis systems such as [16–18], table form documents considered in this paper have a complex structure, and the important point is that the structure has good compatibility with grammatical representation. The proposed grammar in this paper is currently very simple and straightforward, derived from the basic structure of the form document. This implies that the documents which are constructed only with the document structure considered in Sect. 2.2 are accepted by the proposed grammar. This means that at least one of the results produced by a parser with the grammar is correct. However, there still remains a problem that some results of a parser may not be acceptable by the user. To reject such results, more discussion on the grammar itself is necessary.

The grammar proposed in this paper is not the graph grammar which can represent 2D data. Therefore, when we use our grammar for the document analysis, we need to handle both Hlist and Vlist data structure, and also, the grammar is modified to accept first column and row of the bidirectional structure. However, we can easily extend the grammar to the graph grammar which can parse the document more strictly. Note that the parser of graph grammar tends to become very complex; however, there are researches on efficient grammar representation [22] and parser which can be used in the document analysis [23]. When we think of using the grammar for the form correction, this strict parser should be necessary.

When using this method for analysis of real documents, one problem is the process of box type classification. As the document analysis is based on the box type, misclassification can lead to a large analysis error. From our examination, we can classify most of the entry boxes (`ENT`) automatically. This is because half of the entry boxes are empty, i.e. they have no preprinted characters inside. The other half of the entry boxes have preprinted characters, however. In most case, these characters are located at the side of each box. Therefore, we can easily distinguish whether the box is an entry box or not. For the remainder of the boxes which have preprinted characters inside, they are indication boxes (`IND`), in most cases. Therefore, we can easily obtain a small error rate for the box type classification. However, as the system requires completely correct box type classification results, our automatic box-type classification research needs more advance for the practical use.

One possibility of the box type classification algorithm is to incorporate POS tagging algorithm. As our system is based on the grammar analysis, most of the natural language processing algorithm can be applied to our system. In this sense, box type classification and POS tagging have identical functions in the analysis system.

The other problem is that there exists form documents which cannot be accepted by the grammar. As the grammar accepts all the documents which can be handled by the system proposed by Watanabe et al. [13], more than 90% of the formal documents can be handled by our grammar. From our examination, documents which cannot be accepted by our grammar are very difficult to recognize even by human recognition. Therefore, we are planning to make a system which rejects this kind of document and recommend form modification which follows the grammar. This may be useful for the form generation system.

6 Conclusion

In this paper, we first described the functions of the elements appearing in the table form from the communication point of view, and proposed a table form description with document structure grammar. The element of the grammar is a box which is a rectangular field formed by the horizontal and vertical rules. The grammar describes the indication structure by the combination of the adjacent box type.

We also proposed a method to analyze the table form documents with the grammar. As the distribution of the elements in the table form document is 2D, we proposed an analysis system which uses a two box adjacency list which represents horizontal and vertical adjacency of the boxes. Experimental results were described to show the effectiveness of our method.

References

1. Lopresti, D., Nagy, G.: A tabular survey of automated table processing. In: Proceedings of GREC, LNCS 1941, pp. 93–120 (2000)
2. Zanibbi, R., Blostein, D., Cordy, J.R.: A survey of table recognition Models, observations, transformations, and inferences. Int. J. Document Anal. Recogn. (online) (2004)
3. Tang, Y.Y., Ma, H., Liu, J., Li, B.F., Xi, D.: Multiresolution analysis in extraction of reference lines from documents with gray level background. IEEE Trans. Pattern Anal. Mach. Intell. **19**(8), 921–925 (1997)
4. Yu, B., Jain, A.K.: A generic system for form dropout. IEEE Trans. Pattern Anal. Mach. Intell. **18**(11), 1127–1134 (1996)
5. Liu, J., Jain, A.K.: Image-based form document retrieval. In: Proceedings of ICPR, pp. 626–628 (1998)
6. Liu, J., Ding, X., Wu, Y.: Description and recognition of form and automated form data entry. In: Proceedings of ICDAR, pp. 579–582 (1995)
7. Shimotsuji, S., Asano, M.: Form identification based on cell structure. In: Proceedings of ICPR, pp. 793–797 (1996)
8. Hirayama, Y.: Analyzing form images by using lineshared-adjacent cell relations. In: Proceedings of ICPR, pp. 768–772 (1996)
9. Lin, J., Lee, C., Chen, Z.: Identification of business forms using relationships between adjacent frames. Mach.Vision Appl. **9**, 56–64 (1996)
10. Duygulu, P., Atalay, V., Dincel, E.: A heuristic algorithm for hierarchicalrepresen tation of form documents. In: Proceedings of ICPR, pp. 929–931 (1998)
11. Duygulu, P., Atalay, V.: A hierarchical representation of form documents for identification and retrieval. Int. J. Document Anal. Recogn. **5**, 17–27 (2002)
12. Ishitani, Y.: Flexible and robust model matching based on association graph for form image understanding. Pattern Anal. Appl. **3**, 104–119 (2000)
13. Watanabe, T., Luo, Q., Sugie, N.: Layout recognition of multi-kinds of table-form documents. IEEE Trans. Pattern Anal. Mach. Intell. **17**(4), 432–445 (1995)
14. Cesarini, F., Gori, M., Marinai, S., Soda, G.: INFORMys: A flexible invoice-like form-reader system. IEEE Trans. Pattern Anal. Mach. Intell. **20**(7), 730–745 (1998)
15. Bing, L., Zao, J., Hong, Z., Ostgathe, T.: New method for logical structure extraction of form document image. SPIE Proc. **3651**, 183–193 (1999)
16. Rahgozar, M., Cooperman, R.: A graph-based table recognition system. SPIE Proc. **2660**, 192–203 (1996)
17. Cracknell, C., Downton, A.C., Du, L.: An object-oriented form description language and approach to handwritten form processing. In: Proceedings of ICDAR, pp. 180–184 (1997)
18. Belaid, A.: Recognition of table of contents for electronic library consulting. Int. J. Document Anal. Recogn. **4**, 35–45 (2001)
19. Information processing Text and office systems—Standard Generalized Markup Language (SGML). ISO 8879:1986
20. DMA The Document Management Alliance. Available at http://www.infonuovo.com/dma/
21. Information technology Open Document Architecture (ODA) and interchange format: Document structures. ISO/IEC, 8613-2: 1995
22. Amano, A., Asada, N.: Graph grammar based analysis system of complex table form document. In: Proceedings of ICDAR, pp. 916–920 (2003)
23. Costagliola, G., Chang, S., Tomita, M.: Parsing 2D Languages by a Pictorial GLR parser. In: Proceedings of the International Workshop on Advanced Visual Interfaces, pp. 27–29 (1992)