An Adaptative Recognition System Using a Table Description Language for Hierarchical Table Structures in Archival Documents

Isaac Martinat, Bertrand Coüasnon, and Jean Camillerapp

IRISA / INSA, Campus universitaire de Beaulieu, F-35042 Rennes Cedex, France
Isaac.Martinat@irisa.fr, Bertrand.Couasnon@irisa.fr,
Jean.Camillerapp@insa-rennes.fr

Abstract. Archival documents are difficult to recognize because they are often damaged. Moreover, variations between documents are important even for documents having a priori the same structure. A recognition system to overcome these difficulties requires external knowledge. Therefore we present a recognition system using a user description. To use table descriptions in analyzing the image, our system uses the intersections of two rulings with a close extremity of one or each of these two rulings. We present some results to show how our system can recognize tables with a general description and how it can deal with noise with a more precise description.

Keywords: archival documents, table structure analysis, knowledge specification.

1 Introduction

Many works were carried out on table recognition [1], but very few have been carried out on tables from archival documents. These documents are difficult to analyze because they are often damaged due to their age and conservation. The rulings can be broken and skewed or curved. Another difficulty is that paper is thin, ink bleeds through the paper, thus rulings of flip side can be visible. That is why these tables are very difficult to recognize. We want also to recognize sets of documents with a same logical structure whose physical structure can change from one page to the next. To overcome these difficulties, a recognition system needs to have a priori knowledge. Therefore we propose a recognition system using a user description from a language. This language allows to define the logical and physical structures of the tables. The advantage of our language is to describe in the same specification a logical structure with important variations in physical structures (figures 1 and 5). In this paper, we will first present the related work on table representations and on archival document recognition. In section 3 we propose a language to describe tables. Section 4 explains how our recognition system works and uses table descriptions. Before concluding our work, we will show with some results that our system can recognize very different kinds of tables with a same general description and can also recognize noisy and very damaged tables with a more precise description and we validated our system on 7783 images.

W. Liu, J. Lladós, and J.-M. Ogier (Eds.): GREC 2007, LNCS 5046, pp. 9–20, 2008.

2 Related Work

2.1 Table Representations for Editing

Wang [2] proposed a model for editing tables, which is composed of a logical part and a physical part. The logical part contains row and column hierarchy. In the physical part, for a cell or a set of cells, the user specifies the separator type, size, content display (like font size, alignment), ... When a user edits a table, the number of columns and rows must be known. Many other descriptions for tables exist in different languages such as XML but they are for generating tables. For editing, a description must be complete for data, the number of cells is fixed. For table recognition we need to have one description for a set of tables that can have variations between them like the number of columns and rows or the hierarchy.

2.2 Archival Document Recognition

Many works were carried out on table recognition [1] but very few on damaged tables in archival documents. The analysis of these documents is difficult because they are quite damaged. For the recognition of tables with rulings, Tubbs et al. analyzed 1910 U.S. census tables [3] but coordinates for each cell of the tables are given by manually typing an input of 1,451 file lines. The drawback of this method is the long time spent by the user to define this description and the recognized documents can not have physical variations. Nielson et al. [4] recognized tables whose rows and columns are separated by rulings. Projection profiles are used to identify rulings. For each document a mesh is created, and individual meshes are combined to form a template with a single mesh. Individual meshes must be almost identical to be combined, so this method can not recognize documents with important variations between them.

For other archival document recognition methods, a graphical interface is used to recognize archive biological cards [5], lists of Word War II [6]. Esposito et al. [7] designed a document processing system *WISDOM++* for some specific archival documents (articles, registration card) and the result of this analysis can be modified by the user. Training observations are generated from these user operations. All these methods are used for a very specific type of document and the information given by the user is very precise. To help the archival document recognition, systems use a user description [3,8], a graphical interface [5,6], information of other documents of the same type [4] or user corrections [7]. All these methods use external knowledge. However, the table definition process is often quite long and too precise, so these systems do not allow important variations between documents.

We presented in [8] a specific description system for military forms of the 19th Century. We also showed that a general system was not able to recognize these archival documents. This specific description took a long time to write, therefore it is necessary to have a faster way of describing tables. In [9] we presented a table recognition system using a short user description but this system was limited, the row and column numbers were fixed. Furthermore the row and column hierarchies could not be described. Therefore, we propose a general table recognition system for tables using a table description language which can be adapted to damaged archival tables with the introduction of a more precise description.

3 Table Language for Table Recognition

A table is a set of cells organized with columns and rows. We want to recognize the organization of a table, which means locating the cells of a table and labeling each cell with the name and the hierarchy of its column and the name and the hierarchy of its row. We also have to detect table structures from very damaged documents. To solve these two difficulties, we need to use a user description.

3.1 Specification Precision Levels

The language we propose is composed of two parts. The first one is a logical part which describes the row and column hierarchy. The second one is a physical part which allows to specify the row and column separators, and optionally also allows to define the number and/or size of columns and/or rows. The advantage of this language is to describe tables with different levels of precision. On the one hand, the description can be very genesral. In this case, documents with important differences can be recognized with the same description but documents to recognize can not contain noise. For example, for a general description, a multi-row hierarchy can be described without specifying the number of rows for each level. On the other hand, the description given by the user can be very precise. In this case, very damaged documents can be recognized but for the same description, variations between documents can not be important. For example, for a precise description, the numbers of rows and columns can be specified. For a more precise description, sizes can also be given for some columns and some rows. The user can change easily and quickly from a general description to a precise one by adding or modifying some specifications with different precisions as in figure 1. This language also allows to specify a general and precise description, for example the description can be precise for the columns where the number is fixed, and general for the rows where the number is unfixed.

3.2 Table Language Definition

We will now use the term *element* rather than column or row. We propose a language like Wang's model, composed of two parts, a logical one and a physical one. The main differences between our language and Wang's model is that our language allows to specify for a table an unfixed number of columns and rows. In the logical part, the user describes element hierarchy (`COL, ROW`) and the relationship between columns and rows (`COLS_IN_ROW`). The physical part is optional, it allows to specify the number of repetition times for an element (`REPEAT`, `REPEAT+` if the number is unfixed), the size of an element, the separator types (`SEPCOL, SEPROW`). The user can also describe specific separators for some cells (`SEPCELL`).

3.3 Language Examples

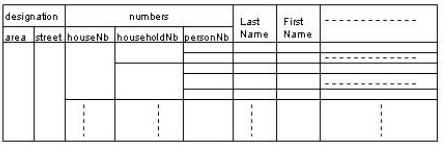
These examples (figure 1) show that a description is easy and fast to write. The words in capital letters are reserved words of the language. To modify the general description to a more precise description, `REPEAT+(1,info)` is replaced by `REPEAT(7,info)`

```
//logical part
//column hierarchy
   COL cols = designation, numbers, info .
   COL designation = area, street .
   COL numbers = houseNb, householdNb, personNb.

//row hierarchy
   ROW rows = boxHead, house.
   ROW boxHead = head1, head2.
   ROW house = household.
   ROW household = person.

//Logical relationships between cols and rows and
//specification for each row the columns that are contained in.
   COLS_IN_ROW head1 = designation, numbers.
   COLS_IN_ROW head2 = area, street , house , household , person.
   COLS_IN_ROW house = houseNb, householdNb, personNb, info.
   COLS_IN_ROW household = householdNb, personNb, info.
   COLS_IN_ROW person = person, info.
   COLS_IN_ROW _ = designation, numbers, info.

 //Physical part
//columns separated by rulings represented by |
//rows separated by rulings represented by -(---------)
   SEPCOL cols = | designation | numbers | REPEAT+(1,info | ) .
   SEPCOL designation = area | street .
   SEPCOL numbers = houseNb | householdNb | personNb.

   SEPROW rows =
     -(-------------------------------)
          boxHead
     -(-------------------------------)
          REPEAT+(1,
               house
     -(-------------------------------)    ).

   SEPROW boxHead =
          head1
     -(--------------------)
          head2  .

   SEPROW house =
        REPEAT+(1,
           household
     -(-------------------------) ).

   SEPROW household =
        REPEAT+(1,
           person
     -(-------------------------) ).
```

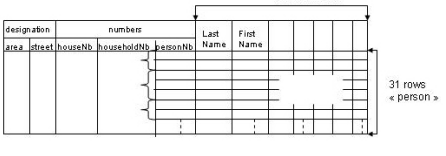
(a) General Description : It is used to recognize very different tables (figures 4 and 5) with an unfixed number of columns and rows at each level of hierarchy.

```
//logical part
//column hierarchy
   COL cols = designation, numbers, info .
   COL designation = area, street .
   COL numbers = houseNb, householdNb, personNb.

//row hierarchy
   ROW rows = boxHead, house.
   ROW boxHead = head1, head2.
   ROW house = household.
   ROW household = person.

//Logical relationships between cols and rows and
//specification for each row the columns that are contained in.
   COLS_IN_ROW head1 = designation, numbers.
   COLS_IN_ROW head2 = area, street , house , household , person.
   COLS_IN_ROW house = houseNb, householdNb, personNb, info.
   COLS_IN_ROW household = householdNb, personNb, info.
   COLS_IN_ROW person = person, info.
   COLS_IN_ROW _ = designation, numbers, info.

 //Physical part
//columns separated by rulings represented by |
//rows separated by rulings represented by -(---------)
   SEPCOL cols = | designation |  numbers | REPEAT(7,info | ) .
   SEPCOL designation = area | street .
   SEPCOL numbers = houseNb | householdNb | personNb.

   SEPROW rows =
     -(-------------------------------)
          boxHead
     -(-------------------------------)
          REPEAT+(1,
               house
     -(-------------------------------)    ).

   SEPROW boxHead =
          head1
     -(--------------------)
          head2  .

   SEPROW house =
        REPEAT+(1,
           household
     -(-------------------------) ).

   SEPROW household =
        REPEAT(31,
           person
     -(-------------------------) ).
```

(b) Precise Description : It is used to recognize very damaged documents (figure 6).



(c) Example of a table which can be recognized with the general description.



(d) Example of a damaged table which can be recognized with the precise description.

Fig. 1. Example of a general description to recognize very different tables wich can be easily modified to a precise description to recognize very damaged documents

and `REPEAT+(1,person)` by `REPEAT(31,person)`. With the general description, the recognition system can recognize tables with important variations between them (figures 4 and 5). Indeed the number of columns is unfixed in this description and for the rows, at each level of hierarchy, the number of rows is unknown. The precise description allows the recognition system to recognize very damaged documents (figure 6). For documents where reverse side rulings are visible, the user can give again a more precise description in giving the approximative sizes for rows and columns. The sizes help the system to avoid detecting reverse side rulings.

4 From the Description to the Image

4.1 Final Intersections

From the image, we extract a set of line segments. Our goal is to match the image information with column and row information given by the language. Therefore we need to associate each line segment with a row or a column separator. We also need to have an intermediate level with common elements to match the image information and the user description. These elements must also be stable. To detect row and column separator within a hierarchy, we need to use line segment extremities. We need to use elements that can be derived from a user description, and these elements must easily be extracted from the image. We propose to define a specific type of intersection, called a *final intersection* which is an intersection involving at least one line segment extremity. From the user description, we can derive the *final intersections* that must be found in the image, and from the image we can extract the final intersections. More specifically, we

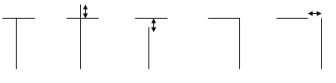


Fig. 2. Examples of *final intersections*, double arrows represent the intersection tolerance

define *final intersection* (figure 2) as an intersection of two rulings with the extremity of one or both of these rulings in close proximity to the other ruling. This definition includes the possibility that the two rulings may not intersect each other. In this case we define *intersection tolerance* as the distance between the two rulings. These *final intersections* allow to detect beginning and end of separators or specific changes in separator types. We do not use cross intersections because these intersections are too ambiguous. The final intersections have stronger dependencies: these intersections are typed and can be differentiated. For example some intersections can be differentiated as a table corner or as the beginning of a row separator.

4.2 Recognition System Using Final Intersections

To detect the table structure, our system perfoms an in-depth analysis of rows and for each Terminal *Row*:

- detects an horizontal Separator we call *SepH*
- from the Table Description : gets the final intersections associated with this *Row* we call *DescrInterList*
- from the Image : gets the final intersections associated with the *SepH* we call ImageInterList
- matchs the *DescrInterList* and *ImageInterList* :
  - if it succeeds, the vertical separators associated with the image final intersections, are labeled (and detected) with the column names from the table description.
  - If this step fails, this matching is *delayed*, which means it will be run later. When the matching is released, that is to say it is run, the detection of column separators during the delay can allow the matching to succeed. The search of intersections is also extended, the intersection tolerance is automatically increased to help the matching to succeed.

The matching succeeds when the final intersections from the description are found in the image. For example, from the description if the number of intersections which must be found in the image equals the number of final intersections in the image, this matching succeeds else it fails.

4.3 System Adaptation in Function of Description Level

When a table description is precise, the system can adapt to a document using the table description, so it can recognize very noisy documents. If a table description is very general, the system will search for final intersections in the image with a small intersection tolerance, the initial value. When a table description is more precise, if after the first detection the system has not detected in the image a structure matching with the table description, the delayed row detections are released. After this release, the intersection tolerance is automatically increased to help the system to find the right structure. For example, if the number of columns is fixed, when the intersection tolerance is increased, the system searches for final intersections in larger zones and can then detect the right number of column separators. When a table description is very precise, sizes for rows and/or columns are given, the system then searches rulings in image zones delimited by these sizes. It helps the system to avoid detecting false rulings, for example the reverse side rulings.

5 Results

The system takes 14 seconds on linux with a 2.0 Ghz processor to recognize an image of 2500x3800 at 256 dpi.

5.1 Example on a Noisy Document

We will show on one synthetic example how our system can recognize noisy documents by using our language. In this first example (fig. 3), the language allows to specify that the document is a table containing 3 columns (A,B and C) and 3 rows. He specifies

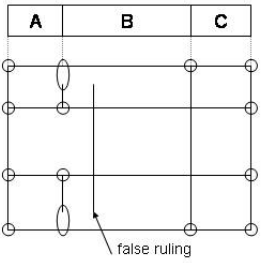
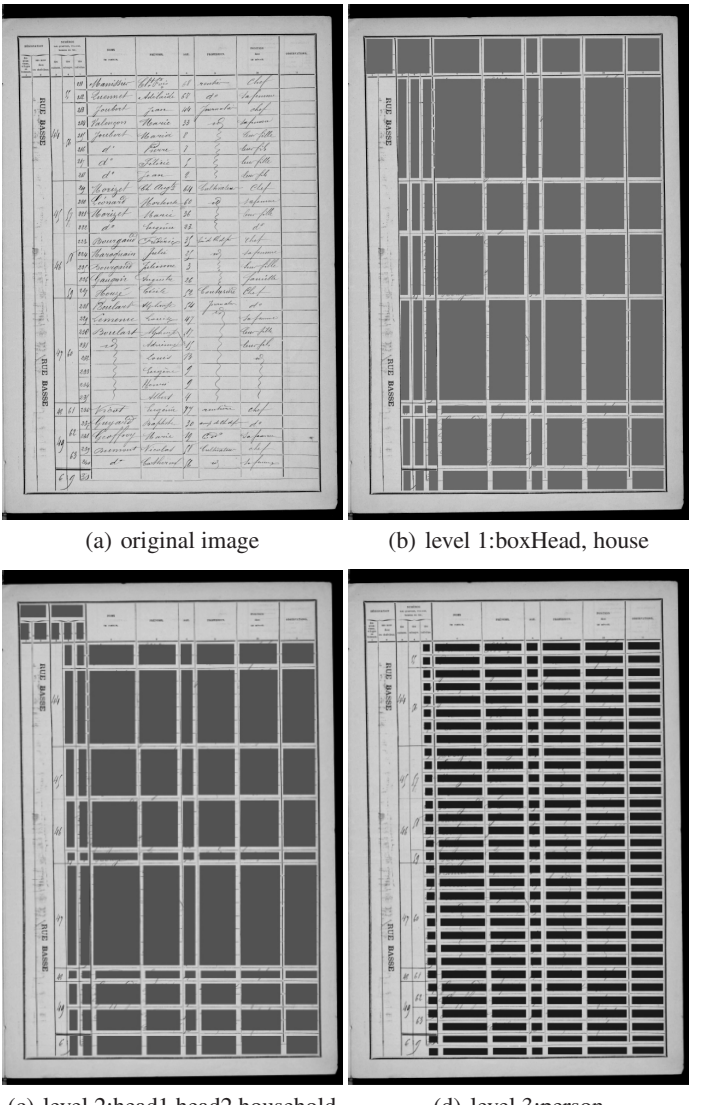


Fig. 3. Example where our system can detect a noisy document with a false ruling. Circles and ellipses represent final intersections.

also that in the second row, the separator of the column A is blank. For the analysis, a preliminary step derives from the table description the final intersections that the system must find in the image. The final intersections in this example are the circles and the ellipses on the image (fig. 3). The system starts the table recognition with the first horizontal separator, extracts from the image the final intersections of the top separator. The system extracts 3 final intersections, although with the description it would have to detect 4 final intersections, so it *delays* this matching. The system then detects the two following horizontal separators as well as the final intersections from each separator and the matching with the description succeeds. The system detects the bottom separator and as the separator for the column A is already detected, the system detects in the prolongation of this vertical separator a final intersection with a higher value of intersection tolerance. After this detection, the *delay* is released (algorithm presented in 4.2), so the system starts to detect the top separator and the final intersections associated with this separator with a higher value of intersection tolerance and as with the bottom separator, it detects the correct final intersections. This example shows how our system can recognize difficult documents. The description allows the system to eliminate false separators, and to detect separators with missing parts.

5.2 General Description

With the same general description (figure 1), the system can recognize census tables from different years (figures 4 and 5) with different structures. On figure 4, the 1881 table contains 8 columns whereas the 1911 table contains 10 columns. For a same year, the row hierarchy is different for each document, thus it is not possible to have a precise description for this hierarchy. The recognition system using this description, after the *boxHead* detection, labeled the column separators in using names from the description. For the row detection, an horizontal separator is detected, then the system gets the vertical ruling that intersects with the left extremity of the horizontal separator. From the terminal level of hierarchy, the system checks if the label of this vertical ruling matches with the specification of the row level. If this checking fails, the system tries again with the upper level of hierarchy until it finds the right level.



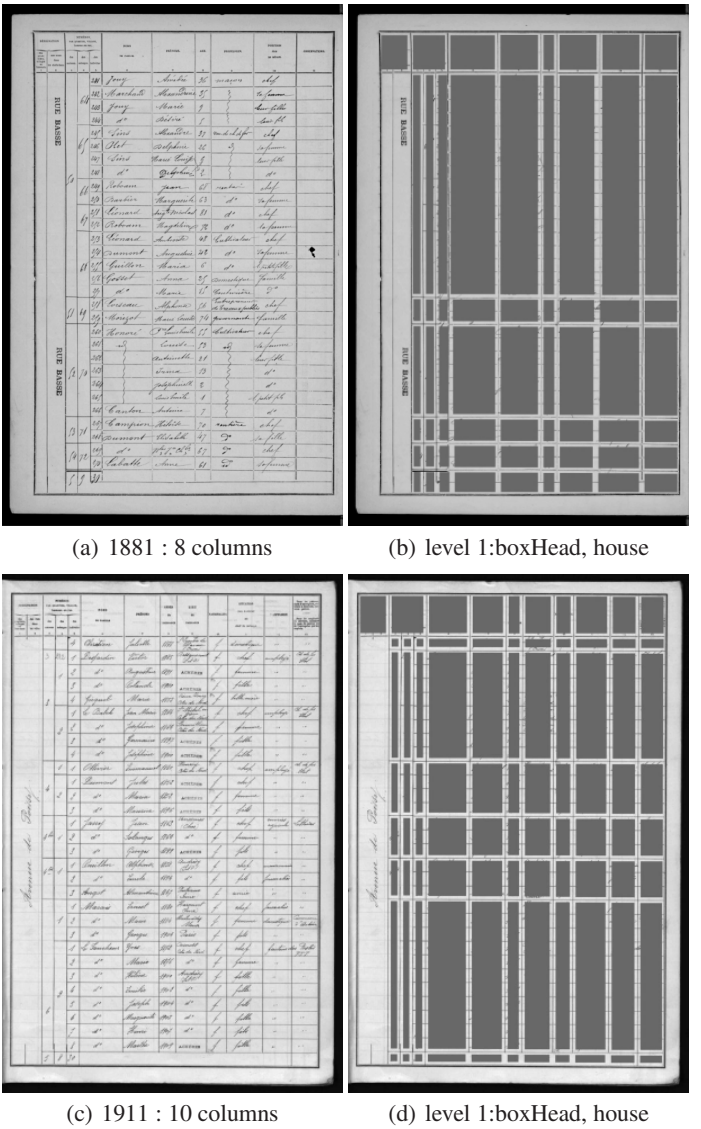
(a) original image (b) level 1:boxHead, house

(c) level 2:head1,head2,household (d) level 3:person

Fig. 4. Census Table of 1881 and the recognized structure with a general description (fig. 1), column number is unfixed like row number at each level of hierarchy

5.3 Precise Description

With a more precise description, the row and column numbers are fixed and the system can recognize the damaged archival document in figure 6. The row hierarchy is not detected but for the lowest level, the *person* rows are detected. As the system fails

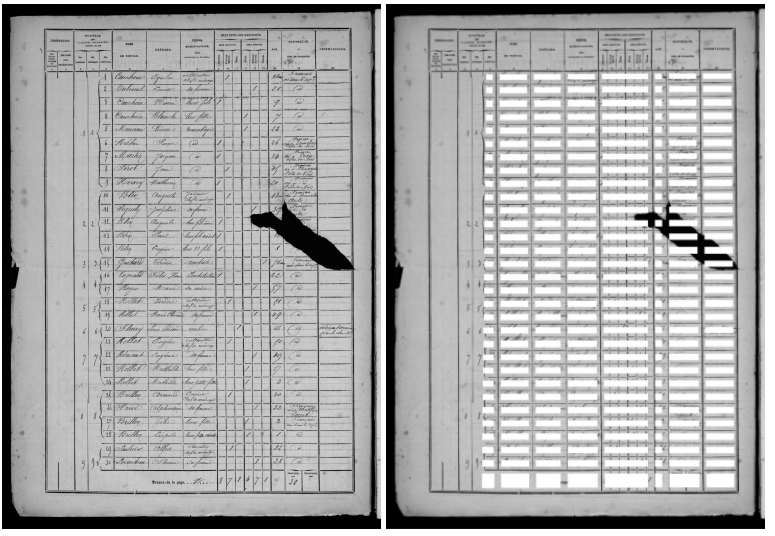


(a) 1881 : 8 columns (b) level 1:boxHead, house

(c) 1911 : 10 columns (d) level 1:boxHead, house

Fig. 5. Census Table of 1881 and 1911 and the recognized structures with the same general description (fig. 1), column number is unfixed like row number

to recognize structures at the first step, the intersection tolerance for ruling gaps as with final intersections is automatically increased until it recognizes the right structure. Therefore the system detects the right structure.



(a) original image (b) level 3:person

Fig. 6. Damaged census table of 1876 and the recognized structure with a precise description, in which the number of rows and columns is fixed

Table 1. Results with the descriptions of the figure 1

level	number of tables	percentage of tables without any error	number of cells	number of detected cells	percentage of detected cells	rejected documents
general description						
all levels	30	66%	11,222	10,925	97,35%	0%
street	30	96%	160	140	87.5%	0%
house, fig. 4(b)	30	93%	2160	2088	96.67%	0%
household, fig. 4(c)	30	73%	2392	2264	94.64%	0%
person, fig. 4(d)	30	83%	6510	6433	98.81%	0%
precise description						
person, figure 4(d)	30	100%	6510	6510	100%	0%

5.4 Statistical Results

Hierarchical Tables. We tested 30 images with the general description of figure 1. These images each contain one table from the 1881 census. All of the code used by the system is totally presented in this figure to which we added an upper level in row hierarchy, a *street* level. Therefore the description specifies now the following row levels : *street*, *house*, *household* and *person*. Indeed for each table (figure 4), the data are presented by street, a street containing several houses, a house containing several households and a household containing serveral persons. These documents can be quite

damaged. 10 images are detected with errors but there are few errors on each image. Therefore, to correctly recognize hierarchical structures, the documents must not be very damaged. However, the lowest levels of hierarchies in damaged documents can be correctly recognized with a more precise description. Table 1 shows the results with the precise description of figure 1 and all the images are well recognized. With this precise description, we have only the lowest level of hierarchy but we get 100% recognition for this level. The precision of the description allows the system to recognize more damaged documents. We did not have time to test on a larger dataset for this kind of table since we must build the groundtruth data manually. In a future work, we will enlarge this dataset.

However, we can test on a larger number of documents with precise and constant descriptions. These descriptions are very precise and contain enough information to assume that if a table is detected, then all of the cells are correctly recognized.

Large quantity of documents. For our first results on a large number of documents we ran our system with precise descriptions on tables without hierarchy. We would have wanted to test our system with fixed hierarchical tables but we did not have this kind of table. Therefore, we detect only the person level on these tables. These results are on two different sets of documents with a precise description for each set. Each set corresponds to one year of census. Table 5.4 shows the results for two years, 1831 and 1836. The description for each year contains the numbers of rows and columns as well as the sizes for each column and each row. In a future work we will study why the system can not recognize the rejected documents. One of the reasons can be the weak contrast value of certain images and the system can fail to detect rulings with weak contrast.

Table 2. Results on a large quantity of documents

year	number of tables	percentage of tables without any error	number of cells	number of detected cells	percentage of detected cells	rejected documents
1831	2722	92.32%	359,304	331,716	92.32%	7.68%
1836	5031	92.72%	950,859	881,685	92.72%	7.28%

6 Conclusion

We presented a language to describe tables. With the same language, table descriptions can be very precise for damaged document recognition as well as very general to detect tables with important variations between them. Moreover, these descriptions can be written quickly. To match table description and image information, we have shown the interest in using some specific intersections which we defined as final intersections. Finally, we have shown through our results how our system can detect a multi-level row hierarchy table with a general description. With this description an important number of different structures can be recognized. If documents are too damaged to be recognized with this description, the user can easily and quickly add or modify some specifications to get a more precise description. The system can then detect very damaged documents,

which is important for the automatic processing of archival documents. We validated our system on 7753 images with a precise description and we got 92.52% recognition. In a future work, we will test on a larger dataset with a general description and we will try to decrease the number of rejected documents using a precise description.

Acknowledgments

This work has been done in cooperation with the *Archives départementales des Yvelines* in France, with the support of the *Conseil Général des Yvelines.*

References

1. Zanibbi, R., Blostein, D., Cordy, J.R.: A survey of table recognition. International Journal of Document Analysis and Recognition (IJDAR) 7(1), 1–16 (2004)
2. Wang, X.: Tabular abstraction, editing, and formatting. PhD thesis, University of Waterloo (1996)
3. Tubbs, K., Embley, D.: Recognizing records from the extracted cells of microfilm tables. In: ACM Symposium on Document Engineering, pp. 149–156 (2002)
4. Nielson, H., Barrett, W.: Consensus-based table form recognition. In: 7th International Conference on Document Analysis and Recognition (ICDAR 2003), Edinburgh, UK, pp. 906–910 (August 2003)
5. He, J., Downton, A.C.: User-assisted archive document image analysis for digital library construction. In: 7th International Conference on Document Analysis and Recognition (ICDAR 2003), Edinburgh, UK, pp. 498–502 (August 2003)
6. Antonacopoulos, A., Karatzas, D.: Document image analysis for world war 2 personal records. In: 1st International Workshop on Document Image Analysis for Libraries (DIAL 2004), Palo Alto, CA, USA, pp. 336–341 (January 2004)
7. Esposito, F., Malerba, D., Semeraro, G., Ferilli, S., Altamura, O., Basile, T.M.A., Berardi, M., Ceci, M., Mauro, N.D.: Machine learning methods for automatically processing historical documents: From paper acquisition to xml transformation. In: 1st International Workshop on Document Image Analysis for Libraries (DIAL 2004), Palo Alto, CA, USA, pp. 328–335 (January 2004)
8. Coüasnon, B.: Dmos, a generic document recognition method: Application to table structure analysis in a general and in a specific way. International Journal of Document Analysis and Recognition (IJDAR) 8(2-3), 111–122 (2006)
9. Martinat, I., Coüasnon, B.: A minimal and sufficient way of introducing external knowledge for table recognition in archival documents. In: Liu, W., Lladós, J. (eds.) GREC 2005. LNCS, vol. 3926, pp. 206–217. Springer, Heidelberg (2006)