Automatic Table Ground Truth Generation and A Background-analysis-based Table Structure Extraction Method

Yalin Wang† Ihsin T. Phillips‡ Robert Haralick†

† Department of Electrical Engineering
University of Washington Seattle, WA 98195 U.S.A.

‡ Department of Computer Science, Queens College
CUNY Flushing, NY 11367 U.S.A.

{ylwang@u.washington.edu}

Abstract

In this paper, we first describe an automatic table ground truth generation system which can efficiently generate a large amount of accurate table ground truth suitable for the development of table detection algorithms. Then a novel background-analysis-based, coarse-to-fine table identification algorithm and an X-Y cut table decomposition algorithm are described. We discuss an experimental protocol to evaluate the table detection algorithms. For a total of $1,125$ *document pages having* 518 *table entities and a total of* $10,941$ *cell entities, our table detection algorithm takes line, word segmentation results as input and obtains around* 90% *cell correct detection rates.*

1 Introduction

Given a document image, document layout analysis specifies the physical embodiment of the image content. Since table is a popular and efficient document element type, table structure extraction is an important problem in the document layout analysis field. Its application can be found in image-XML(eXtensible Markup Language) conversion [1], information retrieval, and document classification, etc.

Many table structure extraction algorithms were addressed [2]-[5]. Some of them were based on predefined table layout structure [2] or relied on complex heuristics which were based on local analysis [3]. A dynamic programming table recognition algorithm was given in [4]. It detected tables based on computing an optimal partitioning of a document into some number of tables. Because it is ASCII(American Standard Code for Information Interchange) text based, it cannot fully make use of document image information when applied to document images and its assumption of using single text column as input is relatively restrictive. A table structure recognition algorithm was reported in [5]. First hierarchical clustering was used to identify columns and then spatial and lexical criteria were used to classify headers. All of the existing algorithms were evaluated on their in-house data set. No general table ground truth data set is publicly available.

We developed an automatic table ground truthing system. It can analyze any given table ground truth and generate documents having similar table elements while adding more variety to both table and non-table parts. Ground truthing is tedious and time-consuming. Using our novel content matching ground truthing idea, the table ground truth data for the generated table elements become available with little manual work. We make this software package publicly available at [18].

Although some background analysis techniques can be found in the literature([6],[7]), none of them, to our knowledge, has been used in the table identification problem. In our table detection algorithm, a preprocessing algorithm is first used to label the column style of a given page and make some modifications to the line and word detection results. Second, a statistical, background-analysis-based, coarse-to-fine table identification algorithm was used to identify table regions. Finally, an X-Y cut table decomposition algorithm was used to obtain the detailed table structure. To systematically evaluate and optimize the algorithms, an area-overlapping performance evaluation method was used for table structure extraction evaluation. Our table detection algorithm was evaluated on a total of $1,125$ document pages having 518 tables and a total of $10,941$ cell entities. The final cell correct detection rates were around 90%.

This paper is organized as follows. In Section 2, we report our automatic table ground truth generation system. We give our table detection algorithm in Section 3. Our per-

Proceedings of the Sixth International Conference on Document Analysis and Recognition (ICDAR'01)
0-7695-1263-1/01 $10.00 © 2001 IEEE

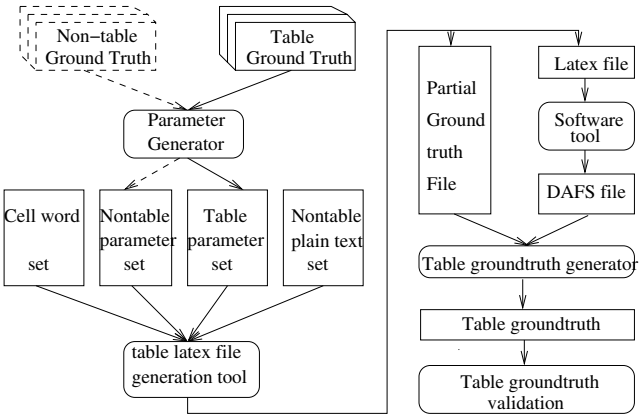


Figure 1. Illustrates automatic table ground truth generation procedure.

formance evaluation method and experimental results are reported in Section 4. We conclude the paper by giving our future work directions in Section 5.

2 Automatic Table Ground Truth Generation

Many of the existing table detection algorithms were developed on a trial-and-error method. Little effort was placed on systematically evaluating the performance of table detection algorithms. The main reason is the lack of a large amount of publicly available accurate table ground truth data to train and test the algorithms. Because manually generating document ground truth proved to be very costly, an automatic, general, accurate and fast table ground truth generation tool is required.

We developed an automatic table ground truth generation system which analyzes any given table ground truth data and generates unlimited document images. In the images, there are tables similar to the given tables but with a controlled variety.

Figure 1 shows the diagram of the system. The following parts describe the automatic table ground truth generation procedure.

Parameter Generator: This software is used to analyze a given table ground truth and non-table ground truth. Two kinds of parameter sets, $\mathcal{T}$ and $\mathcal{N}$, are designed. There are 12 table layout parameters in $\mathcal{T}$, e.g. column justification, spanning cell position, etc. There are 4 non-table layout parameters in $\mathcal{N}$. e.g. text column number, if there is marginal note, etc. Clearly, $\mathcal{T}$ is designed to add more variety to table instances and test the mis-detection performance of any table detection algorithm. Parameter set $\mathcal{N}$ is designed to add more variety to non-table instances and test the false alarm performance of any table detection algorithm. Currently, the part which automatically estimates non-table parameters has not been implemented, so we enclose them in dashed lines in Figure 1(a).

Table Latex File Generation Tool: This software randomly selects two parameter elements from sets $\mathcal{T}$ and $\mathcal{N}$. The resulting parameter for a page is a reasonable element in $\mathcal{T} \times \mathcal{N}$. We precomputed two content sets $\mathcal{C}$, $\mathcal{P}$. They are cell word set and non-table plain text set. Elements of $\mathcal{C}$ are random, meaningless English character strings. Elements of $\mathcal{P}$ are the text ground truth file from UW CDROM III [8]. Sets $\mathcal{C}, \mathcal{P}$ are the contents of table entities and non-table entities in the generated LaTeX [9] file, respectively. We make sure every element in $\mathcal{C}$ is unique in both $\mathcal{C}$ and $\mathcal{P}$ and it can only be used once for a given file. This software writes out two files: a LaTeX file and a partial ground truth file. In the partial ground truth file, there are table, row header, column header and cell entities with their content and attributes such as cell starting/ending column number, etc.

DAFS File Generation Tools Several software tools are used and some minimum manual work is required in this step. LaTeX turned the LaTeX files into DVI(DeVice Independent) files. The DVI2TIFF software [10] converts DVI file to a TIFF(Tagged Image File Format) file and a so-called character ground truth file which contains the bounding box coordinates, the type and size of the font, and the ASCII code for every individual character in the image. The CHARTRU2DAFS software [18] combines each TIFF file and its character ground truth file and converts it to a DAFS(Document Attribute Format Specification) file [11]. The DAFS file has content ground truth for every glyph, which is the basis of content matching in the next step. Then line segmentation and word segmentation software [13] [14] segments word entities from DAFS file. Since we cannot guarantee a 100% word segmentation accuracy, a minimum of manual work using Illuminator [12] tool is required to fix any incorrect word segmentation results inside tables.

Table Ground Truth Generator: Since we know every word in the tables appears once, we can use content matching method to locate any table related entity of interest. Our software tries to locate any word contents from partial ground truth file in DAFS file. If not, an error is reported. Here is the way to make the previous step even simple. We only need run table ground truth generator twice. The only places we need look at are the files with some errors in the first run. After the correction, we run this software again to obtain the final table ground truth data.

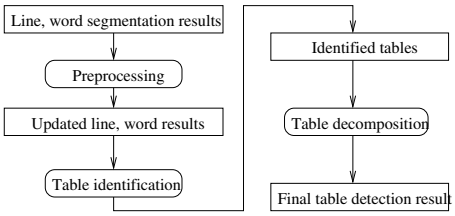


Figure 2. The process diagram of the table detection algorithm

Table Ground Truth Validation: For normal ground truthing work, validation is a required step to make sure that we get correct ground truth. Our table ground truth validation is also automatically done. It checks the geometric relations among table, row, column and cell entities. If there is any discrepancy, the page can be either removed or given to further manual checking.

3 Table Detection Algorithm

Table detection problem includes two subproblems: table identification and table decomposition. The goal of table identification is to separate table regions from non-table regions in a given page. Since table itself is a hierarchical structure, table decomposition techniques are used to determine the structure of a given table and identify its elements such as row/column headers, cells, etc.

Figure 2 gives an overview of our table detection algorithm. Input data to our table detection algorithm are the segmented line and word entities with roughly separated regions([13], [14]). Figure 3(a) shows an example of the input image and (b) shows table detection result on the same page.

3.1 Preprocessing

A column style labeling part is used to label the column structure. Assuming the maximum number of columns is two in our data set, we designed a column style labeling algorithm which can label a given page by one of the three column styles: double column, single column with marginal note and single or mixed column style.

Our column style classification is based on background analysis technique [15]. To construct the background structure, the foreground entities are words. A vertical blank block $\mathcal{VR} = b_r \times b_c$, with lefttop vertex coordinate (x_{b1}, y_{b1}), is a *blank separator* if and only if it satisfies the following conditions: **(1).** $\frac{b_c}{mw} > \theta$, where mw is the median width of text glyphs in the whole page. Here θ is empirically determined as 3; **(2).** It has the largest row number among all the blank blocks. If there are more than one such blank blocks, one with largest column number is selected. If there is another tie, one of them is randomly selected.

Given blank separator $\mathcal{BK} = b_r \times b_c$, with lefttop vertex coordinate (x_{b1}, y_{b1}), the two features for the page column style classifier are: **(1).** $lr = \frac{b_r}{lv_r}$, where lv_r is the row number of the page live-matter part; **(2).** $pr = \frac{x_{b1}+b_c/2}{x_{lv}+lv_c/2}$, where x_{lv} is the column coordinate of the lefttop vertex of the page live-matter part and lv_c is the column number of the page live-matter part.

Suppose we have C column styles, $d_1, ..., d_C$. We compute the column style c, for a given page, as $P(c = d_j|lr, pr)$, where $j = 1, ..., C$. After we identified the column style, we make adjustments to line and word segmentation results according to the labeled column style.

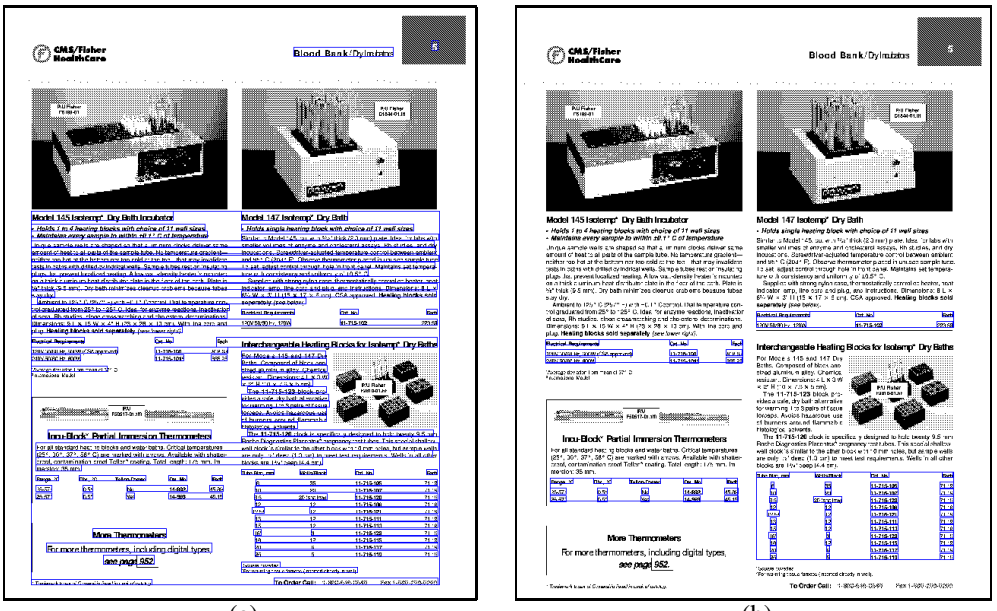
3.2 Table Identification Algorithm

Our table identification algorithm is a coarse-to-fine algorithm. First, we determine table entity candidates by locating the large horizontal blank blocks [15]. Then a statistical based table refinement algorithm is used to validate the table entity candidates and reduce the false alarms.

After we identify large horizontal blank blocks, we group the vertically adjacent large horizontal blank blocks together and then group their horizontally adjacent words together as table entity candidates. Then we use the idea stated in Section 3.3 to decompose the table entities. Clearly, the table candidates have many false alarms among them. A statistical table refinement algorithm is used to validate each table candidate.

For each table candidate, three features are computed.

- Ratio of total large vertical blank block [15] areas over identified table area. Let t be an identified table and $\mathcal{B}$ be the set of large vertical blank blocks in it, $ra = \frac{\sum_{\beta \in \mathcal{B}} Area(\beta)}{Area(t)}$;
- Maximum difference of the cell baselines in a row. Denote the set of the cells in a row i as $\mathcal{RC}_i$, $\mathcal{RC}_i = \{c_{i,1}, c_{i,2}, ..., c_{i,i_m}\}$. Denote the set of $\mathcal{RC}_i$ as $\mathcal{RC}$, $\mathcal{RC} = \{\mathcal{RC}_i, i = 1, ..., m\}$, where m is the row number in the table. Let $baseline(c)$ be the y coordinate of the cell entity bottom line, $mc = \max_{\mathcal{RC}_i \in \mathcal{RC}} (\max_{c_{i,j} \in \mathcal{RC}_i} (baseline(c_{i,j})) - \min_{c_{i,j} \in \mathcal{RC}_i} (baseline(c_{i,j})))$;
- Maximum difference of the justification in a column. Denote the set of cells in a column, i, in the table body region $\mathcal{CC}_i = \{c_{i,1}, c_{i,2}, ..., c_{i,i_n}\}$. Denote the set of $\mathcal{CC}_i$ as $\mathcal{CC}$, $\mathcal{CC} = \{\mathcal{CC}_i, i = 1, ..., n\}$, where n is the column number in the table. Let $x_{i,j}, y_{i,j}, w_{i,j}, h_{i,j}$ represent the bounding box of the cell $c_{i,j} \in \mathcal{CC}_i$. We estimate the justification of a column, $i, i = 1, ..., n$,



(a) (b)

Figure 3. Illustrates input and result of table detection algorithm. (a). An example of input data to table detection algorithm. The graphic parts in the image have been filtered. Segmented line entities are shown. (b). An example of result of table detection algorithm. Table cell and table entities are shown.

		Total	Correct	Splitting	Merging	Mis-False	Spurious
Real Data Set	Ground Truth	679	609 (89.69%)	2 (0.29%)	12 (1.77%)	56 (8.25%)	0 (0.00%)
	Detected	654	609 (93.12%)	4 (0.61%)	6 (0.92%)	35 (5.35%)	0 (0.00%)
Whole Data Set	Ground Truth	10,941	9,882 (90.32%)	267 (2.44%)	321 (2.93%)	461 (4.21%)	10 (0.09%)
	Detected	10,737	9,882 (92.04%)	548 (5.10%)	154 (1.43%)	143 (1.33%)	10 (0.09%)

Table 1. Cell level performance of the table detection algorithm on real data set and whole data set.

by computing the vertical projection of the left, center, and right edge of $c_{i,j}, j = 1, ..., i_n$,

$$C_{left}[i] = \max_{c_{i,j} \in CC_i}(x_{i,j}) - \min_{c_{i,j} \in CC_i}(x_{i,j})$$

$$C_{center}[i] = \max_{c_{i,j} \in CC_i}(x_{i,j} + w_{i,j}/2) - \min_{c_{i,j} \in CC_i}(x_{i,j} + w_{i,j}/2)$$

$$C_{right}[i] = \max_{c_{i,j} \in CC_i}(x_{i,j} + w_{i,j}) - \min_{c_{i,j} \in CC_i}(x_{i,j} + w_{i,j})$$

$$J_i = min\{C_{left}[i], C_{center}[i], C_{right}[i]\}$$

The maximum difference of the justification in a column, mj, is computed as: $mj = max(J_i), i = 1, ..., n$.

Then we can compute the table consistent probability for table t as $P(consistent(t)|ra(t), mc(t), mj(t))$ If $P(consistent(t)|ra(t), mc(t), mj(t)) > 0.5$, we label the table candidate as a table entity.

3.3 Table Decomposition Algorithm

Similar to recursive X-Y cut in [16], we do a vertical projection on the word level in each identified table. Because of the table structure, we can expect the projection result to have peaks and valleys. We can separate each table column which starts from a valley and ends at the next valley. After we construct the table columns, we can get cell structures and their attributes such as starting/ending row, starting/ending column.

4 Experimental Results

Our testing data set has $1,125$ document pages. All of them are machine printed, noise free data. Among them, 565 pages are real data from different business and law

books. Another 560 pages are synthetic data generated using the method stated in Section 2. The parameter files we used in this experiment can be obtained at [18]. A hold-out cross validation experiment [17] was conducted on all the data with $N = 3$. Discrete lookup tables were used to represent the estimated joint and conditional probabilities used at each of the algorithm decision steps.

Suppose we are given two sets $\mathcal{G} = \{G_1, G_2, ..., G_M\}$ for ground-truthed foreground table related entities, e.g. cell entities, and $\mathcal{D} = \{D_1, D_2, ..., D_N\}$ for detected table related entities. The algorithm performance evaluation can be done by solving the correspondence problem between the two sets. Performance metrics developed in [13] can be directly computed in each rectangular layout structure set. In our current system, we only try to decompose the identified tables into cells, so the performance evaluation was only done on cell level. The numbers and percentages of miss, false, correct, splitting, merging and spurious detections on real data set and on the whole data set are shown in Table 1.

5 Conclusion and Future Work

In this paper, we described a system which can automatically generate various table ground truth based on some predefined parameters. These parameters can be estimated from the real table instances. We developed a novel background-analysis-based table detection algorithm. We defined table detection performance evaluation problem as the correspondence between ground truth and detection results on different levels. We conducted the experiments on $1,125$ document pages having 518 table entities and a total of $10,941$ cell entities, the accurate segmentation rates on cell level were around 90% on both real and whole image data sets.

There are many open problems in the table detection field. In the future, we need a better table decomposition algorithm which can generate row header, column header and table body levels. We need make use of our automatic table ground truth generation tool on more real table instances and get greater table variety. We can further generate new table instances whose parameters are estimated from the table parameter set.

References

[1] Y. Wang, I.T. Phillips and R. Haralick, "From Image to SGML/XML Representation: One Method", *DLIA'99*, Bangalore, India, September 1999

[2] J. H. Shamilian, H. S. Baird, and T. L. Wood, "A Retargetable Table Reader", *Proceedings of the 4th ICDAR*, pp. 158-163, Germany, August 1997.

[3] T. G. Kieninger, "Table Structure Recognition Based on Robust Block Segmentation", *Document Recognition V,* pp. 22-32, January 1998.

[4] J. Hu, R. Kashi, D. Lopresti, and G. Wilfong, "Medium-independent table detection", *SPIE Document Recognition and Retrieval VII*, pp. 291-302, San Jose, California, January 2000.

[5] J. Hu, R. Kashi, D. Lopresti, and G. Wilfong, "Table Structure Recognition and Its Evaluation", *SPIE Document Recognition and Retrieval VIII*, San Jose, California, January 2001.

[6] A. Antonacopoulos. Page Segmentation Using the Description of the Background. *Computer Vision and Image Understanding*, pp. 350–369, June 1998.

[7] H. S. Baird. "Background Structure in Document Images", *Document Image Analysis*, pp. 17–34, 1994.

[8] I. Phillips. "Users' Reference Manual", *CD-ROM, UW-III Document Image Database-III*, 1995.

[9] M. Goossens, F. Mittelbach and A. Samarin, "The LaTeX Companinon", *Addison-Wesley Publishing Company*.

[10] T. Kanungo, "DVI2TIFF User Manual", *UW English Document Image Database - (I) Manual*, 1993.

[11] RAF Technology, Inc., "DAFS:Document Attribute Format Specification", 1995.

[12] RAF Technology, Inc., "Illuminator User's Manual", 1995.

[13] J. Liang, "Document Structure Analysis and Performance Evaluation", *Ph.D thesis*, Univ. of Washington, Seattle, WA, 1999.

[14] Y. Wang, I. T. Phillips and R. Haralick, "Statistical-based Approach to Word Segmentation", *15th International Conference on Pattern Recognition, ICPR2000*, Vol. 4, pp.555-558, Barcelona, Spain, September 2000.

[15] Y. Wang, R. Haralick, and I. T. Phillips: "Improvement of zone content classification by using background analysis", *DAS2000*, Rio de Janeiro, Brazil, 10-13 December, 2000.

[16] J. Ha, R. Haralick and I. T. Phillips, "Recursive X-Y Cut using Bounding Boxes of Connected Components", *Proceeding of 3rd ICDAR*, pp. 952-955, 1995.

[17] R. Haralick and L. Shapiro. "Computer and Robot Vision", *Addison Wesley*, Vol 1, 1992.

[18] http://isl.wtc.washington.edu/ylwang/auttabgen.html