Extracting Tabular Information From Text Files

Scott Tupaj, Zhongwen Shi*, Dr. C. Hwa Chang**

EECS Department, Tufts University, Medford, MA 02155

*Hassan Alam***

BCL Computers, 155A Moffett Park Dr., Suite 104, Sunnyvale, CA 94089

**{stupaj, zshi, hchang}@ee.tufts.edu **hassan@shell.portal.com*

Abstract

This paper presents work done in locating and extracting tables and their contents from document images. While most research in the area of table analysis and recognition has focused on analyzing the raster image, our approach builds upon the advances in optical character recognition (OCR) software to preserve the layout of tabular data by means of white space. By using methods to analyze the geometry, syntax, and the semantics of the character data, as well as utilizing some well-known image processing techniques, we are able to 1) isolate embedded tables from documents, and 2) identify table components such as title blocks, table entries, and footer blocks. Furthermore, the table analysis techniques presented in this paper can also be applied when analyzing blocks of text isolated by traditional methods such as connected component analysis[1] or bounding box [2].

1. Introduction

Tables are a means for presenting structured data in paper documents. They provide an efficient method for presenting data in scientific journal articles as well as financial data in annual reports and prospectuses. Most work in table analysis has focused on analyzing the raster image to extract lines and identify significant white space intervals to yield clues as to table location and

surface form. Our work focuses on examining a document image and its content. The content analysis is done by parsing the text output from commercial OCR packages to identify tables and their contents. Many advances in document analysis have allowed OCR software to captures a document's text as well as its format. Analyzing the document on a character level proves much more effective in terms of speed and semantic interpretation than processing raster images.

The algorithm consists of four phases. First, using image processing techniques, the document image is segmented and analyzed to isolate potential table areas. Secondly, these areas are passed to an OCR engine that produces text output. In the third phase, text is analyzed to isolate the beginning and end of the table(s). Finally, local analysis is performed to isolate table components (title blocks, cells, and footers). Figure 1 shows the flow diagram. The dotted paths denote analysis phases that are not described in this paper, but can be incorporated.

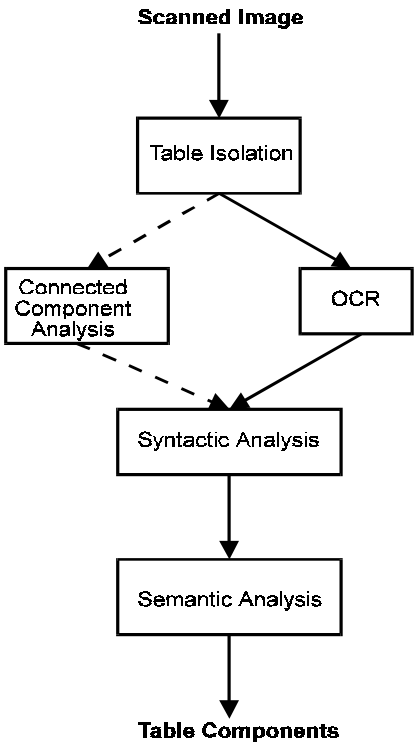


Figure 1. Diagram of the four phases of analysis

This paper presents an overview of our method and our results. Section 2 describes the goals of extracting tables, Section 3 describes types of tables, Section 4 describes our method, Section 5 describes the results, and Section 6 describes areas of future work.

2. Goals

A number of research efforts have been underway to analyze tables in documents. These efforts have focused on different parts of the whole problem. Some have focused on extracting tables in specific domains [3], others have examined the semantics of tables [4], while others have worked on locating closed form tables (line-bounded) on page images [5]. Our goal is primarily to build upon the achievements of OCR software in preserving table structure in terms of space delimited or tab delimited text.

In developing methods to analyze data in this form, we develop techniques that serve as a general model. The methods can later be applied to more specific data sets, such as text block and line locations on an image. The text data we analyze gives us two important pieces of information in one package: 1) the table geometry and 2) the table text. Since the semantics of tables are defined by geometric relations as well semantics of the text that comprises the table, we are able to parse the text data returned from OCR programs and extract information about the table. We wish to be able to isolate tables and their components to enable storage in some higher level format that reflects the semantics of the table (relational databases, spreadsheets, etc.).

3. Overview of Tables

3.1. Table Domains

To develop a broad, robust table analysis system we selected tables from two distinct domains, scientific journal articles and financial reports.

3.1.1. Technical Tables

Scientific journal articles use tables to show results of experiments or demonstrate trend in data. The data types include text and numeric data. In addition to standard numeric data, there is extensive use of specific measuring units such as distance, energy, time, or units specific to an experiment. There is often semantic information embedded in the table that allows multiple entries per cell as well as tables embedded inside tables. Figure 2 shows an example of a scientific table.

TABLE III

DIRECT AND SNEAK SIGNAL ATTENUATION VS FREQUENCY

Frequency	1.55 kMc		1.85 kMc		2.15 kMc	
Module Number	Direct Attenuation (db)	Sneak Attenuation (db)	Direct Attenuation (db)	Sneak Attenuation (db)	Direct Attenuation (db)	Sneak Attenuation (db)
301 to 303	23.5	—	22.8	—	22.7	—
301 to 404	22.5		22.0		23.0	
302 to 301 via 404	—	46.0	—	44.0	—	43.0

Figure 2. A scientific table

3.1.2. Financial Tables

Financial tables are used to convey financial results for companies, mutual funds, and other investment and financial related institutions. Tables include balance sheets, income statements, and annual performance results. The data types include dates, currency, percentage, and text. There is often arithmetic calculation done in the tables that include sum and ratios (percentages). Figure 3 show an example of a financial table.

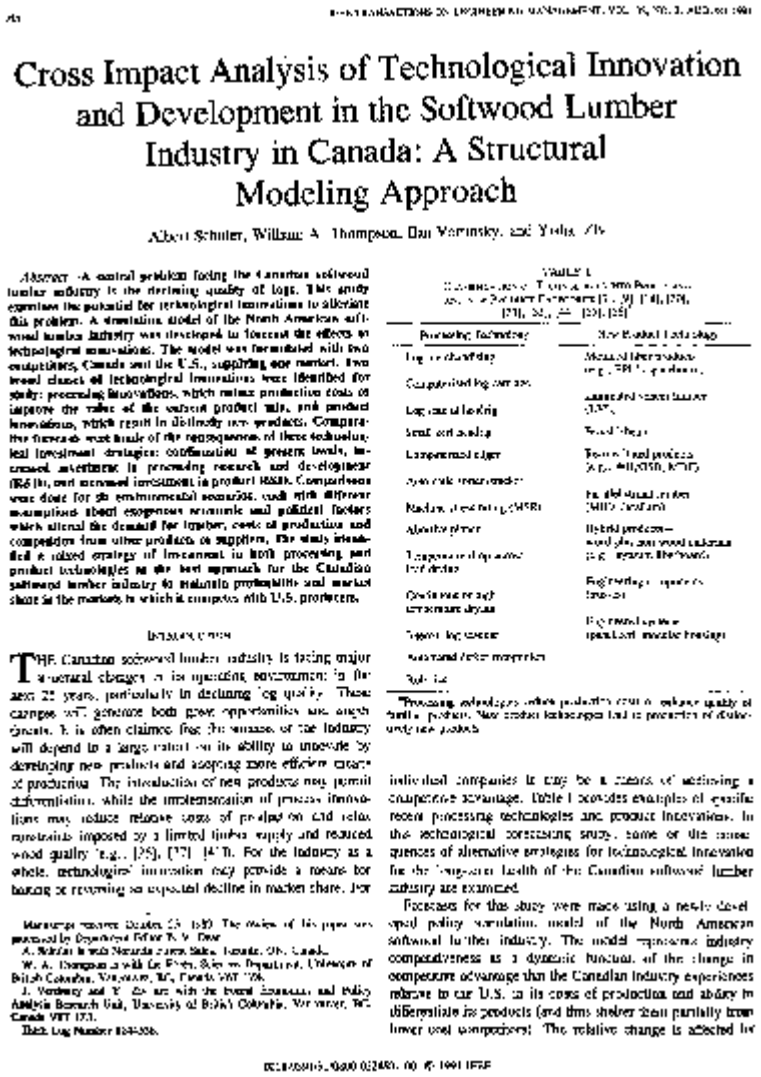
Marlea Company
Comparative Retained Earnings Statement
December 31, 1994 and 1993

	1994	1993	Increase (Decrease*) Amount	Percent
Retained earnings, January 1	$137,500	$100,000	$37,500	37.5%
Net income for the year	91,000	76,500	14,500	19.0%
Total	$228,500	$176,500	$52,000	29.5%
Dividends:				
On preferred stock	$ 9,000	$ 9,000	—	—
On common stock	40,000	30,000	$10,000	33.3%
Total	$ 49,000	$ 39,000	$10,000	25.6%
Retained earnings, December 31	$179,500	$137,500	$42,000	30.5%

Figure 3. A financial table

3.2 Table Locations

To be truly useful, a system for analyzing tables must perform two tasks, locate all tables embedded in a page, and decompose individual tables into their respective elements. Locating a table in on page involves differentiating the tables from other elements such as body text, headings, titles, bibliographies, lists, author listings, abstracts, line drawings, and bitmap graphics. Figure 4 shows an example of a table embedded in document containing some of the elements described. Furthermore, tables may exist on a page by themselves with no other document elements. The analysis system must be able to detect such a scenario and not perform the extraction algorithms.



Cross Impact Analysis of Technological Innovation and Development in the Softwood Lumber Industry in Canada: A Structural Modeling Approach

Figure 4. An embedded table

Variations include tables embedded on two column pages, multiple tables on a page, and tables spanning across two column pages.

3.3. Types of Tables

We considered the following types of tables in our analysis: completely bounded tables (line-defined), partially bounded tables, and unbounded tables. Kojima and Akiyama [6] give a complete description of the different types of tables that usually occur in documents.

4. Method

Our method consists of four phases. Each of the four analysis phases is described in detail below. Our first phase is purely mathematical. The second phase involves the OCR processing. The third phase uses a syntactic approach, and the fourth phase begins the semantic analysis. We find this approach to yield the best results. Each phase, using a higher level of analysis, is able to detect and correct errors by the previous phases. There is also efficiency in terms of speed, as the low level mathematical analysis is done on the largest data set (the raster image), and the time consuming semantic analysis is performed on the text representation of the table body.

4.1. Isolating Table Zones on the Image

In the first level of analysis, we are concerned with isolating potential table areas on a page. These areas will be passed to an OCR package to convert to text form. The process to accomplish this task must take into account the wide variety of documents we wish to process. For example, we do not wish to segment a document containing a single table in the same manner we segment a multiple column journal article containing embedded tables. The segmentation methods should not interfere with any tables that appear on the page. We use a combination of white space analysis and keyword analysis to mark rectangular zones on the page that contain tabular information. This analysis phase has very loose constraints, since the core analysis is done analyzing the text in the area identified as a possible table. We need only isolate areas that *may* contain tabular data. Furthermore, the zones we extract may contain extraneous document data not belonging to the table.

When receiving document image data from a fax or scanner, the image is often imperfect. It may contain noise or more importantly, may be skewed. Skewed images present a problem for document analysis due to the assumption that text flows left to right on the horizontal (0°) axis. We use techniques outlined in [7] to determine the skew angle of the document.

The image processing techniques that we employ focus primarily on analyzing horizontal and vertical projection data. Projections provide a means of transforming two dimensional data into a one dimensional signal. More importantly, once the skew angle of the document is determined, projections can be taken at any angle, thus eliminating the need to deskew the entire document before processing. We use a modification of the recursive X-Y cut method [8] to segment the document. Instead of creating specific parsing rules in regards to the white space relationships, we analyze the projection profile data using statistical methods.

In the first horizontal pass, we wish to find significant vertical white space gaps. These will appear as a series of zero values in the horizontal projection. However, since noise and any horizontal lines may interfere with a series of zero values, we modify the projection technique by taking into account three values:

pixel count(*n*) - the number of black pixels along the scan line.

cross counts(*c*) - the number of times the pixel value changes in the scan line[9].

extent (*e*) - the number of pixels between the first and last black pixel on the scan line

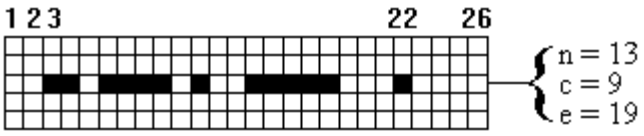


Figure 5. Example of the three projection values

We projection value (*p*) for a given scan line is a function of *n*,*c*, and *e* (*p* = *f*(*n*,*c*,*e*)). The following properties are used to determine *f*.

Bar of noise or horizontal line - high *n*/*e* ratio and low *c*/*n* ratio.

Speckled noise, ascenders or descenders - low *n*/*e* ratio and high c/*n* ratio

Text - high *n*/*e* ratio and high *c*/*n* ratio

Figure 6 shows the effectiveness of this method. Note that p is also set to zero if it falls below some threshold t. (In this example, $t = 5$). White space gaps that are above k standard deviations from the mean are considered significant document separation points. For our analysis we use $k = 1.5$. The horizontal cuts are verified by a rule base. Horizontal cuts that are equally spaced, or create consecutive segments that contain significant white space gaps in the vertical projection, are probably table entry delimiters (Figure 7). These cuts are nullified.

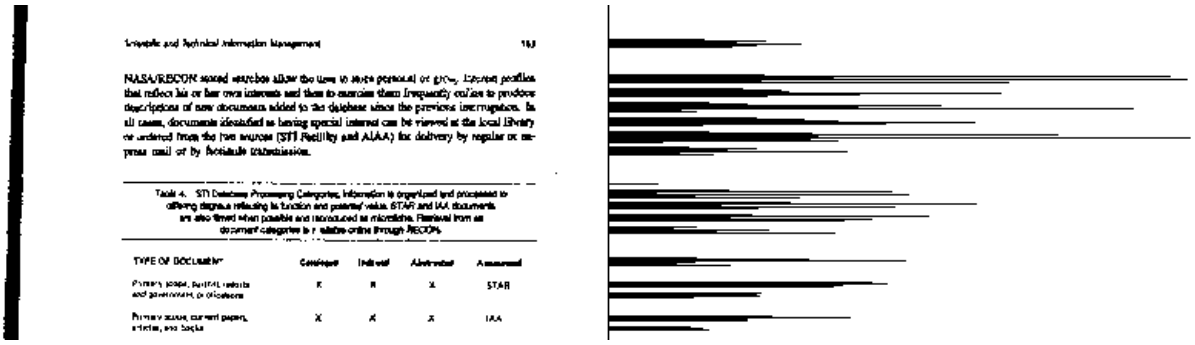


Figure 6. Modified horizontal projection data

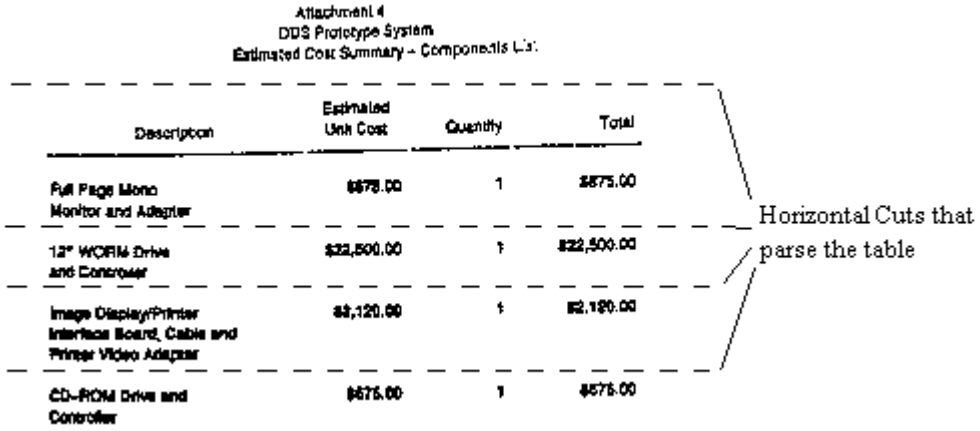


Figure 7. Horizontal cuts that segment a table

Once the document is segmented into valid horizontal strips, we examine the vertical projection of each segment to see whether or not it contains multiple columns of flow text. We use the standard projection technique of simply counting the number of black pixels along a vertical scan line.

Again we use a rule base to validate potential vertical cuts. We assume that no document will have more than three columns. The analysis should also be concerned with not splitting a two or three column table. Usually the text column widths in tables are significantly different, and the white space that separates them will be greater than a half inch. If multiple column structure is identified, the segment is split into two (or three) segments, and the horizontal analysis described above is applied to each segment

We use the following characteristics to identify multiple column structures:

Let each interval of zero values have width Z_i,

Let each interval of non-zero values have width N_i

$\{ Z_1 N_1 Z_2 N_2 Z_3 \}$ exists with

1) $Z_2 < 30$
2) $\dfrac{\min(N_1, N_2)}{\max(N_1, N_2)} < 0.9$

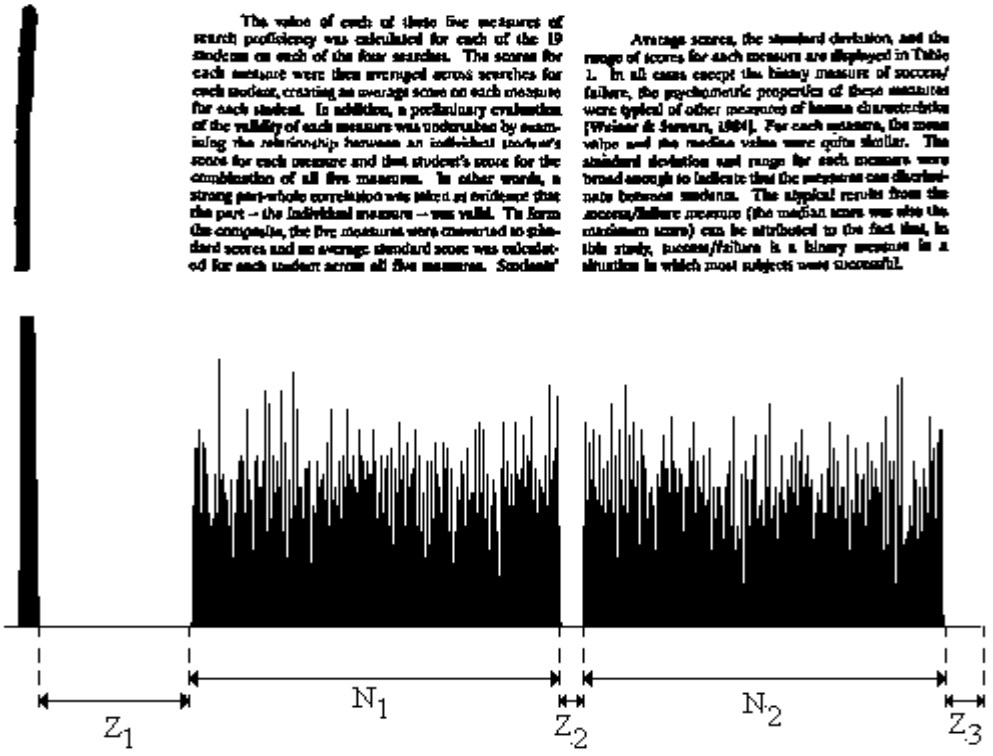


Figure 8. Example of identifying multi-column structure

Once the document image is segmented, each segment is classified as having table or non-table characteristics. A quick analysis of the text in the segment will give clues as to the content. We look for certain keywords, numeric data, and other repeated data types. Each potential table segment is passed through an OCR engine to produce a space delimited text file.

4.2. OCR Engine

Cutting edge OCR technology is able to preserve the table geometry by inserting blank lines and space characters to denote significant semantic separation within the table text. Although there may be some errors, the output provides a good base for preliminary analysis.

4.3. Text File Analysis

Text files are much easier to analyze for two reasons. First, the coordinate system has been reduced from pixel coordinates to text coordinates. Although this is a "lossy" transformation, (i.e. font sizes and lines are lost), it is easier to isolate lines of text and white space without using image processing techniques such as connected component analysis and projection profiles. Second, the words can be analyzed for content and semantic meaning, i.e. data types and keywords can be identified. This compensates for the errors in retaining the surface geometry of the table. It should be noted that with minimal modifications, the methods described here can be used on the image analysis level (Figure 9).

The image pre-processor described above is able to define the left and right boundaries of the table, since embedded tables usually follow the text flow of the page. However, finding the top and bottom boundaries of a table requires both geometric and semantic analysis. The technique we developed is a syntactic approach. Each line of text is classified as one of three types and assigned a token:

B - Blank line
S - Line containing a single text entity
C - Line containing multiple text entities (usually tabbed text)

An *entity* is defined as a set of characters bounded on the left and right by either 4 or more space characters, or the left and/or right table boundary. In the image analysis, this would refer to horizontally aligned text blocks separated by a threshold value (Figure 9).

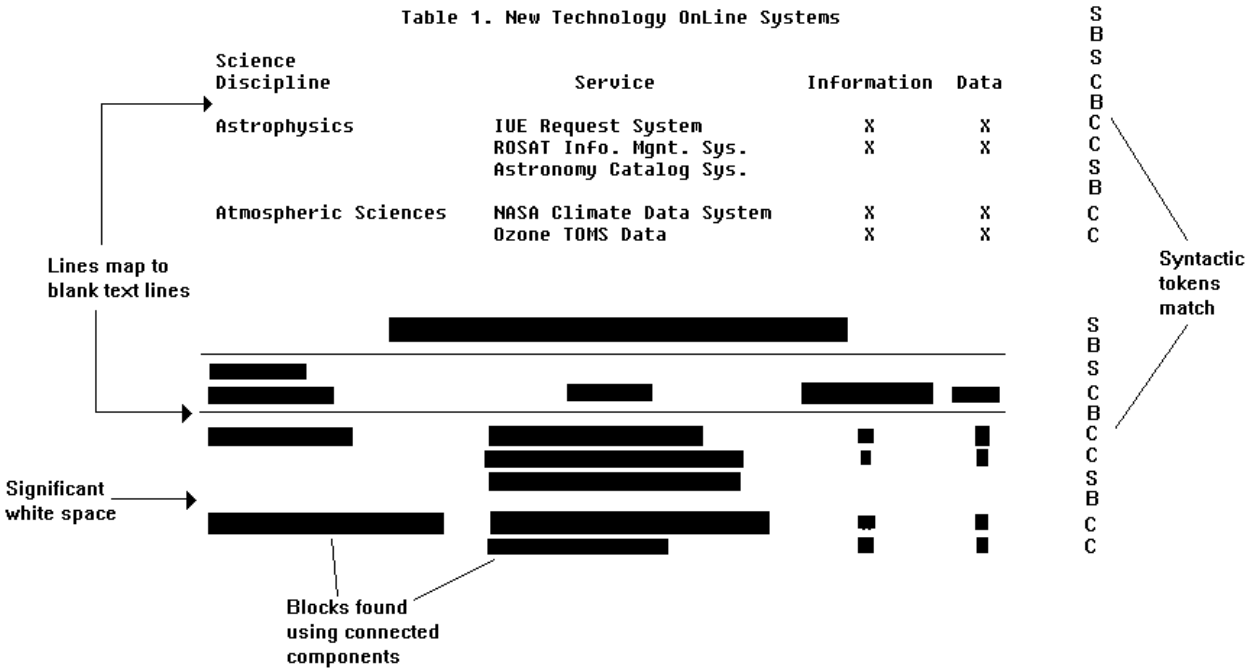


Figure 9. Relationship between ASCII data and image analysis

We start the analysis by finding keywords that may appear in table titles. The line that contains the keyword(s) serves as a starting point. Since most title blocks appear at the top of the tables, we allow a buffer of *n* lines and scan downward for sequences of lines whose tokens match certain sequences. If none of these tokens are found, we assume the title block is below the table and scan upward. The lines whose tokens fit the sequences in Figure 10 are flagged as table body lines. These are appended to the title block.

{C}
{S}
{B S (C ∪ S)}
{B S B S}
{B B C}
{B B (S ∪ B) C}

Figure 10. Sequences of lines that are conducive to table structures

If {B B B (B ∪ S)} is found, then we have reached the end of the table. The table footer is isolated using a keyword rule base and the fact that footers always appear below the table body.

4.4. Isolating Table Components

To identify the table components, we base our method on the techniques used by [10] to divide the table into a Cartesian grid of basic cells and assign spreadsheet-like coordinates to each cell (Figure 11). Multiple line entries and spanning headers are expressed as combinations of these basic cells.

Year of birth	*n*		Age at assessment (years)			
	Boys	Girls	1989	1990	1991	1992
1981	13	15	9	10	11	12
1980	36	48	10	11	12	13
1979	38	40	11	12	13	14
1978	12	14	12	13	14	15
1977	6	4	13	14	15	16
1976	35	35	14	15	16	17
1975	39	33	15	16	17	18
1974	24	37	16	17	18	19
1973	13	9	17	18	19	20

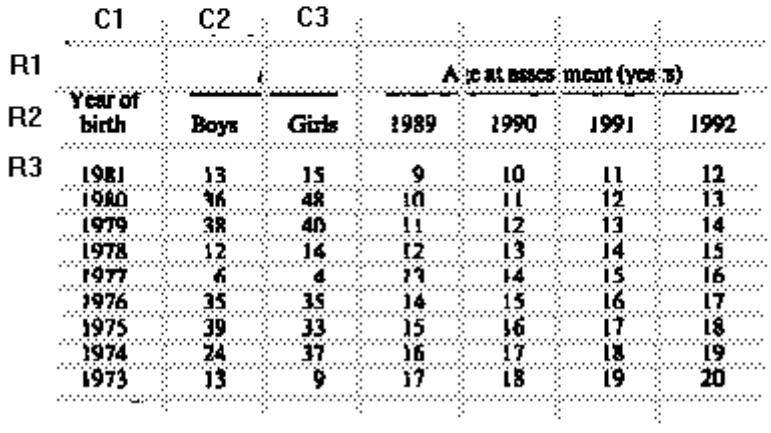


Figure 11. Splitting the table into basic cells

As the lines of text can be used to delimit rows in the table, we are presented with the problem of finding implied vertical separation points in the table. The space intervals on the lines of text help

us form plumb lines, which serve as potential vertical separation points. Our method is based on the notion of intersecting consecutive vectors of space intervals [11]. Figure 12 shows this procedure.

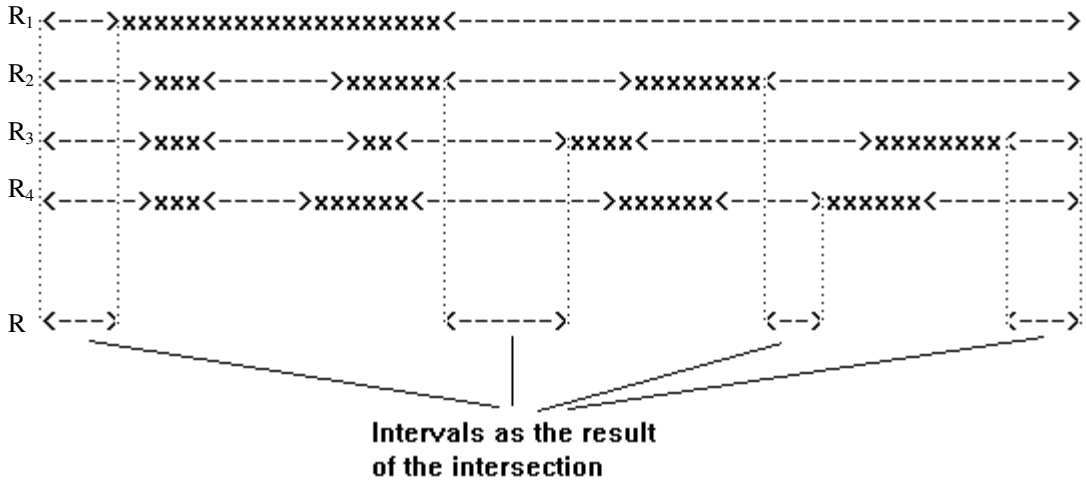


Figure 12. Intervals and their intersection

For each line of text in the table, we create a vector:

$$R_i = \{(a_1,b_1), (a_2,b_2), ..., (a_n,b_n)\}$$

where a_j is the leftmost point of a space interval and b_j is the rightmost point.

We then create vector R:

$$R = R_1 \cap R_2 \cap ... \cap R_m$$

where each R_i is a vector such that $|R_i| > 2$. This denotes lines of the table that contain at least two elements. The midpoint of each space interval in R represents a potential vertical plumb line in the table.

Each table line that contains only two space intervals can be either flow text (header / footer) or a single entry in that row. For each R_i that contains 2 intervals, the $R \cap R_i$ is formed. If $|R'| << |R|$, then a significant number of plumb lines were lost trying to include R_i in the table. Hence, the

table line is flagged as body text. Otherwise, the entry falls in the column structure of the table. In this case, R is set to R' and the line is included in the table body. Figure 13 shows an example of this scenario. To summarize, our technique of using space intervals serves two functions: 1) it separates the table body from the headers and footers and 2) provides vertical separation points for extracting cells.

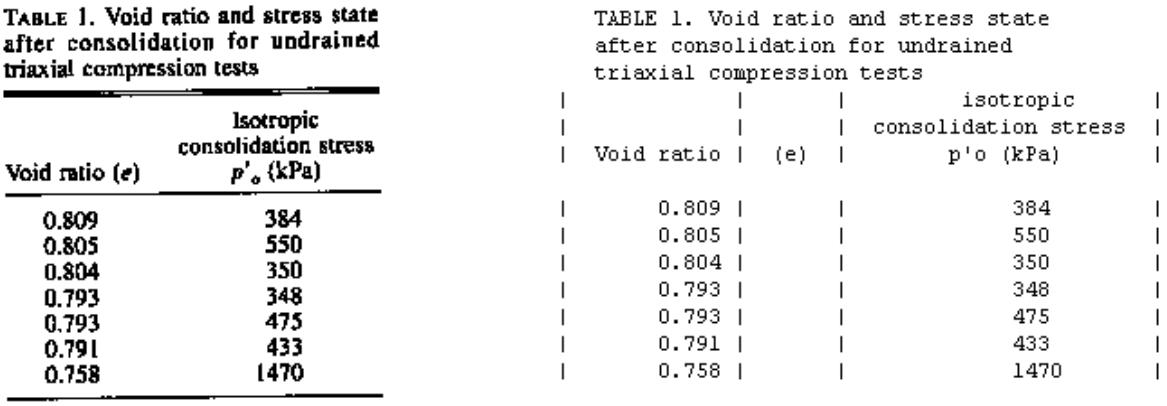


TABLE 1. Void ratio and stress state after consolidation for undrained triaxial compression tests

Void ratio (e)	Isotropic consolidation stress p'_o (kPa)
0.809	384
0.805	550
0.804	350
0.793	348
0.793	475
0.791	433
0.758	1470

TABLE 1. Void ratio and stress state
after consolidation for undrained
triaxial compression tests

Void ratio	(e)	isotropic consolidation stress p'o (kPa)
0.809		384
0.805		550
0.804		350
0.793		348
0.793		475
0.791		433
0.758		1470

```
Table Title:  TABLE 1. Void ratio and stress state after consolidation for undrained triaxial com

Table Footer:

Cell[0][0]: <NULL>
Cell[0][1]: <NULL>
Cell[0][2]: isotropic

Cell[1][0]: <NULL>
Cell[1][1]: <NULL>
Cell[1][2]: consolidation stress

Cell[2][0]: Void ratio
Cell[2][1]: (e)
Cell[2][2]: p'o (kPa)

Cell[3][0]: 0.809
Cell[3][1]: <NULL>
Cell[3][2]: 384
```

Figure 13. Vertical plumb lines determined from space intervals

At this point, further semantic analysis is applied on the array of basic cells. The example above shows two scenarios that occur most in this analysis.

1) A "ghost" column may form due to extra spacing
2) Column hierarchy is not detected in the case of spanning superheaders.

Using a semantic rule base helps adjust for these errors. Consistent data types, and the presence of numerous empty cells are just some of the clues we are looking into to solve such problems.

5. Results

Our work in table analysis can be measured in three areas: accurate detection of tables, accurate detection of exact tables, and accurate detection of table components. Each of these accuracies is measured by precision and recall rates. Precision measures the degree of correct identification, and recall measures the degree of correct identification with respect to the total number of elements.

	Number of Samples	Detection of Tables		Detection of Cells	
		Precision	Recall	Precision	Recall
Scientific Journals	100	0.930	0.924	0.970	0.916
Financial Tables	20	0.714	0.750	0.920	0.890

Figure 14. Results

As can be seen technical tables from scientific journal articles were easier to locate and extract than financial tables. This is because of the relatively uniform method of presenting technical data. While standards exist for financial reports, the layout is substantially more varied and harder to detect with current algorithms.

6. Future Work

The approach outlined in this paper represents only the beginnings of an analysis technique to extract and recognize tables and their components. A number of areas still need work. The output from the OCR engine often produces imperfect spacing and erroneous characters. We therefore need to emphasize more on the semantic analysis of tables by content examination, as well as better topological models for describing the tables, to correct for these errors. Also, in the image processing phase, it would be helpful to include more information for the table analysis phase to use in its analysis, such as line locations and more specific text locations, font sizes, etc., that the OCR output cannot retain. A number of efforts are underway in this area, and we expect to see improvements in the future.

7. Bibliography

[1] Fletcher L.A. and Kasturi R., "A Robust Algorithm for Text String Separation from Mixed Text/Graphics Images". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 10, pp.910-918, 1988.

[2] Mahadevan, U and Nagabushnam, R.C, "Gap metrics for word separation in handwritten lines" *Proc. of the 3rd Intl. Conf. on Document Analysis and Recognition*, Vol. 1, pp.124-127, 1995.

[3] Howaldt, K., Maderlechner, G., Scherl, W. "Structural interpretation of tables in scanned office documents" *6th Scandinavian Conference on Image Analysis*, pp. 839-842, 1989.

[4] Douglas, S., Hurst, M., Quinn, D. "Using natural language processing for identifying and interpreting tables in plain text" *Fourth Annual Symposium on Document Analysis and Information Retrieval*. pp. 535-546

[5] Balasubramanian, S., *et al*. "Information extraction from tabular drawings" *SPIE Vol. 2181 Document Recognition*, pp.152-163, 1994.

[6] Kojima, H., Akiyama, T., "Table recognition for automated document entry system" *SPIE Vol. 1384 High Speed Inspection Architectures, Barcoding, and Character Recognition*, pp. 285-292, 1990.

[7] Srihari, S. and Govindaraju, V., "Analysis of textual images using the Hough transform" *Machine Vision and Applications*, Vol. 2, pp. 141-153, 1989.

[8] Nagy, G., and Seth, S., "Hierarchical representation of optically scanned documents," *Proc. of 7th Int. Conf. on Pattern Recognition*, pp. 347-349, 1984.

[9] Akiyama, T., and Hagita, N., "Automated entry system for printed documents," *Pattern Recognition*, Vol. 23, No. 11, pp. 1141-1153, 1990.

[10] Laurentini, A. and Viada, P. "Identifying and understanding tabular material in compound documents" *Proceedings of the Intl. Conf. on Pattern Recognition*, pp. 405-409, 1992.

[11] Rahgozar, M., *et. al*. "Tabular document recognition" *SPIE Vol. 2181 Document Recognition*, pp 87-96, 1994.

[12] Hori, O. and Doermann, D.S. "Robust table-form structure analysis based on box-driven reasoning". *Proc. of the Third International Conference on Document Analysis and Recognition*. Vol. 1, pp. 218-221, 1995

[13] Watanabe, T. and Fukumura, T. "A framework for validating recognized results in understanding table-form document images". *Proc. of the Third International Conference on Document Analysis and Recognition*. Vol. 2, pp. 536-539, 1995

[14] Alam, H. *et al.*. "Extracting tables from printed documents". (*To be published*), 1995