International Journal on Document Analysis and Recognition manuscript No.
(will be inserted by the editor)

A Survey of Table Recognition:

Models, Observations, Transformations, and Inferences

Richard Zanibbi, Dorothea Blostein, and James R. Cordy

School of Computing, Queen's University, Kingston, Ontario, Canada, K7L 3N6
E-mail: {zanibbi,blostein,cordy}@cs.queensu.ca

Received May 29, 2003 / Revised October 28, 2003

Abstract. Table characteristics vary widely. Consequently, a great variety of computational approaches have been applied to table recognition. In this survey, the table recognition literature is presented as an interaction of table models, observations, transformations, and inferences. A table model defines the physical and logical structure of tables; the model is used to detect tables, and to analyze and decompose the detected tables. Observations perform feature measurements and data lookup, transformations alter or restructure data, and inferences generate and test hypotheses. This presentation clarifies the decisions made by a table recognizer, and the assumptions and inferencing techniques that underlie these decisions.

Key words: Table recognition – Modelling table structure – Performance evaluation – Information retrieval

1 Introduction

Many documents contain tables that could be recovered for reuse, compression, editing, and information retrieval purposes. In table recognition, a definition of table location and composition (a table model) is used to recover tables from encoded documents. Hu et. al.[39] have termed the two main sub-tasks of table recognition table detection and table structure recognition. In table detection, instances of a table model are segmented. In table structure recognition, detected tables are analyzed and decomposed using the table model. In this paper we break down table detection and structure recognition further, describing both as sequences of three basic operations: observations, transformations, and inferences. Observations include feature measurements and data lookup, transformations are operations that alter or restructure data, and inferences generate and test hypotheses (e.g. table locations).

Correspondence to: Richard Zanibbi

In this survey we present the table recognition literature in terms of the interaction of table models, observations, transformations, and inferences, as presented in Figure 1. Surveys that take other views of the literature are also available[27,58,59]. We use the view presented in Figure 1 to assist us in answering important questions about the decisions made by table recognizers: what decisions are made? What is assumed when decisions are made? What inferencing techniques make decisions? On what data are decisions based? Table models play a crucial role in the decision making process, as they define which structures are sought after and define or imply a set of assumptions about table locations and structure. Input parameters define decision thresholds and tolerances, provide values for table model parameters, and may include additional information used in decision making.

In the remainder of this paper we define tables and describe table models used in recognition (Section 2), outline the observations, transformations and inferences used by different systems for table detection and structure recognition (Sections 3, 4, and 5), address performance evaluation methods used to determine the sufficiency of a table recognizer for a specific recognition task (Section 6), and finally identify open problems and conclude in Section 7.

2 Table Models

Tables are one of the visualizations people use to search and compare data[55]. More specifically, they visualize indexing schemes for relations, which may be understood as a set of $n-$tuples where n is the number of sets in the relation[21]. The sets of a relation underlying a table are called domains or dimensions. A relation may be presented many different ways in a table. Dimensions may be laid out in different row and column arrangements, repeated, or ordered in various ways. The arrangement of dimensions in a table affects which data are most easily accessed and compared[17,26,55].

The parts of a table are described in Figure 2. Dimensions of a relation whose elements are to be searched and

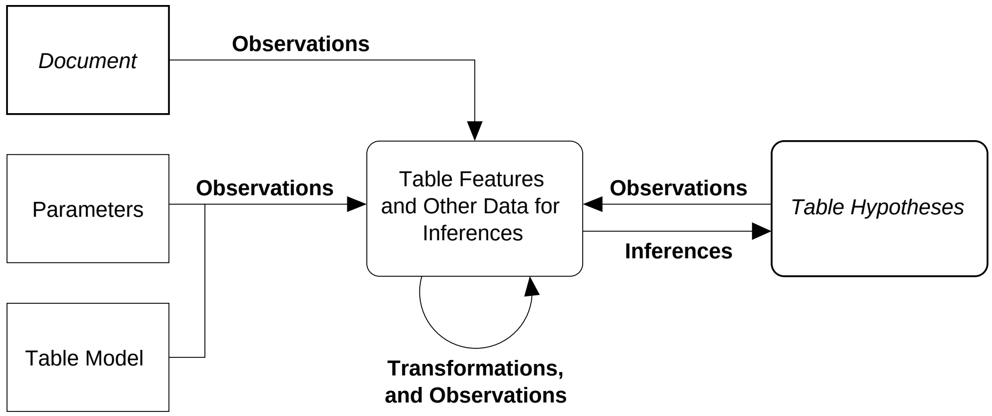

Fig. 1. The Table Recognition Process. A table model defines the structures that a table recognizer searches for. Table recognizers detect and decompose tables using observations, transformations, and inferences. Inferences generate and test table location and structure hypotheses. Observations provide the data used by inferences; these are feature measurements and data lookups performed on the input document, table model, input parameters, and existing features and hypotheses. Transformations of features permit additional observations. Input parameters define or constrain the table model, observations, transformations, and inferences of a table recognizer.

compared in a table have their elements located in the body; the names and elements of remaining dimensions are placed in the boxhead and stub as headers, which are then used to index elements located in the body. The stubhead may contain a header naming or describing the dimension(s) located in the stub. Often headers are nested to visually relate dimension names and elements, and to factor one dimension by another. For example, the dimension 'school term' is factored by 'year' in the stub of the table in Figure 2; this is indicated by indenting (nesting) the 'school term' elements (e.g. 'Winter') below the 'year' elements (e.g. '1992'). As another example, the header for element 'Ass3' is nested below the name of its associated dimension ('Assignment'). Regions where individual dimension names and elements are located are called cells. A group of contiguous cells in the body is referred to as a block. Cells are separated visually using ruling lines (e.g. the boxhead and stub separators in Figure 2) and whitespace, or tabulation[21]. This results in the familiar arrangement of cells in rows and columns. If the boxhead, stub, and stubhead are absent, we are left with a list or matrix of values; these do not have headers used to index data, which is a defining feature of tables.

In practice individuals often alter or adapt the parts of a table as presented in Figure 2. For example, headers or explanatory text might appear in the body of a table. In some cases, tables even contain tables within cells, or are compositions of tables, producing complicated indexing structures[26,89]. However, the majority of tables studied in the table recognition literature are described well by Figure 2.

Tables also often have associated text, including titles (e.g. Figure 6), captions, data sources (e.g. Figure 3), and footnotes or additional text that elaborate on cells (e.g. Figure 6). The text in a document that cites a table sometimes provides additional information regarding a table's contents[20]. The focus in table recognition so far has been on recovering tables themselves; only a small number of papers have addressed text associated with a table[20,69,73,86].

In this paper we will restrict our discussion to tables that present textual data, as this is the class of tables that have been studied for recognition. This includes tables encoded in text files (e.g. ASCII), such as the one rendered for presentation in Figure 6. Sources of plain text tables include email messages and automated reporting systems (e.g. the EDGAR financial reporting system[51]). The remainder of Section 2 addresses the physical and logical structure of tables, HTML encodings of table structure, and models used in generating and recognizing tables.

2.1 Physical and Logical Structure of Tables

As with all entities sought after in document recognition, tables have physical and logical structure[27,29,62]. For tables, physical structure describes where regions containing parts of tables are located in an image or plain text file (e.g. Figure 6), while logical structure defines the types of these regions and how they form a table. All regions of interest in a table have both physical and logical structure. For example, the location of a line in a table image is part of the physical structure of a table, while the type of a region (in this case, 'line') is part of the logical structure of a table. Similarly, the intersection of two lines is defined in logical structure, while the location of the intersection is defined using geometry (part of physical structure).

We define the most abstract level of logical structure for tables to be the indexing scheme from headers to cells

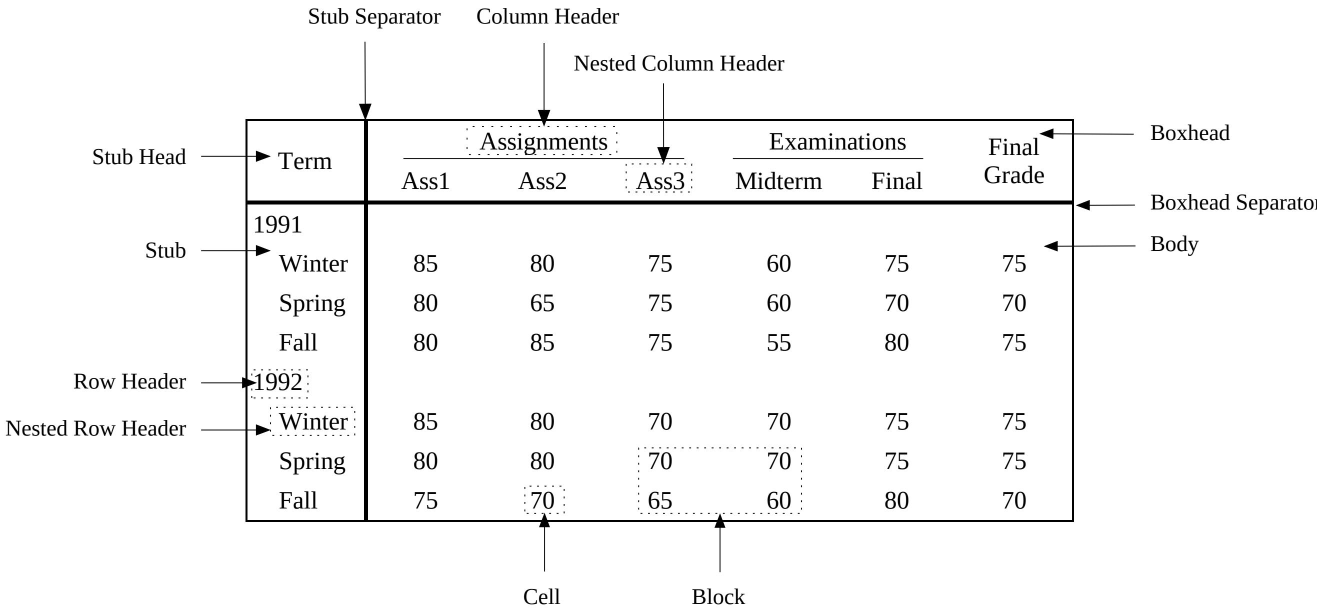

Fig. 2. Table Anatomy. The example and terms shown here are taken from Wang[89], where terminology from the Chicago Manual of Style[23] is used. Though not shown in this figure, tables often have associated text regions, such as a title, footnotes, or the source for table data.

located in the body of a table. This defines a relation describing a table, but it may not be minimal (e.g. the table may repeat dimensions of a relation). Defining the minimal relation underlying a table requires knowledge about the subject matter domain of the table, such as how dimensions are related, fixed ranges for particular dimensions, or synonyms for dimension names. This is information that tables themselves do not provide: their function is to visualize an indexing scheme for data in a relation, not to interpret the relation in the data domain. As a result, we consider anything more abstract than the table's indexing structure to be part of a table's subject matter domain rather than part of the logical structure of the table itself.

For a table encoding, the least abstract level of logical structure describes the type of the smallest regions of interest. In an image this might be a connected component. For plain text files, this might be be regions of adjacent non-whitespace character (connected components of characters in the text file). For HTML files, this might be a tagged cell. At this level there is no relational structure, only primitive regions with types.

Intermediate levels of logical structure describe the composition of smaller regions into larger ones, and relate regions to one another. For example, a series of connected components may be joined into a line (relating the connected components), which is found to intersect with another line (relating the two lines). An important intermediate level of logical structure describes cell adjacencies, or topology. Cell topology is often described using a table grid. Table grids are formed by extending all line and whitespace cell separators to the edges of a table region (see Figure 4). The grid allows indexing cell locations using a scheme similar to those in spreadsheets, in which columns and rows are enumerated. Depending on the structure of a table, grid locations may be empty, or cells may span multiple grid locations as in Figure 4.

The physical structure of a table can be encoded in a text or image file, while logical structure may be encoded using a markup language such as HTML (see Figure 5). The tags in the markup language describe data types and relations (i.e. they define a graph on labelled data). In HTML, tags can be used to define the table grid (contents and relative positions of cells), types of separator (lines vs. whitespace), and the location of the body, header (boxhead), and footer areas. HTML does not encode the stub location, indexing structure, or underlying relation of a table.

In practice, tables encoded in HTML often do not use the header and footer tags, and use tags that are not part of the table tag set (for an example, see Figure 5). Also, the table environment is often used to layout lists and matrices of data in a grid, with no indexing by headers[42, 91].

2.2 A Table Model for Generation: Wang's Model

The most complete table model in the literature was designed to support generating table images from logical structure descriptions by Wang[89]. Wang's model separates table structure into three parts: an abstract indexing relation, a topology defining the placement and ordering of dimensions within the boxhead, stub, or both regions, and formatting attributes which include fonts, separator types, and layout constraints. Formatting attributes are associated with logical structures: dimensions and their elements, table regions (e.g. body or stub),

Journal	Full Name	Details	
		Appears	Publisher
TPAMI	IEEE Transactions on Pattern Analysis and Machine Intelligence	monthly	IEEE
IJDAR	International Journal on Document Analysis and Recognition	quarterly	Springer-Verlag
PR	Pattern Recognition	monthly	Elsevier
IJPRAI	International Journal on Pattern Recognition and Artificial Intelligence	eight times/year	World Scientific

Source: from a listing of pattern recognition journals provided online at http://www.ph.tn.tudelft.nl/PRInfo/

Fig. 3. A Table Describing Document Recognition Journals. This fully ruled table was rendered by an HTML viewer using the source file shown in Figure 5. Note the text below the table indicating the source of the data presented in the table.

A1−A2	B1−B2	C1−D1	
		C2	D2
A3	B3	C3	D3
A4	B4	C4	D4
A5	B5	C5	D5
A6	B6	C6	D6

Fig. 4. Grid Describing the Location of Cells for the Table in Figure 3. Table grids are formed by extending all separators to the border of a table. Here rows are represented by numbers, and columns by letters. The table in Figure 3 has a 'Manhattan' layout, in which all separators meet at right angles. For non-Manhattan layouts cells may not be rectangle-shaped (and the table grid more complex as a result). Occasionally cells occupy more than one location in the table grid: these are called spanning cells. The topmost cells in Figure 3 are all spanning cells, located at grid locations A1-A2, B1-B2 and C1-D1.

```
<TABLE RULES=ALL BORDER=1 CELLPADDING=5 ALIGN=CENTER>
  <THEAD>
    <TR>
      <TD ROWSPAN=2>Journal</TD>
      <TD ROWSPAN=2 ALIGN=CENTER>Full Name</TD>
      <TD COLSPAN=2 ALIGN=CENTER>Details</TD>
    </TR>
    <TR>
      <TD>Appears</TD>
      <TD>Publisher</TD>
    </TR>
  </THEAD>
  <TBODY>
    <TR>
      <TD>TPAMI</TD>
      <TD>IEEE Transactions on Pattern Analysis
          and Machine Intelligence</TD>
      <TD>monthly</TD>
      <TD>IEEE</TD>
    </TR>
    ...
  </TBODY>
</TABLE>

<P ALIGN=CENTER>
  Source: from a listing of pattern recognition
  journals provided online at
  http://www.ph.tn.tudelft.nl/PRInfo/
</P>
```

Fig. 5. Partial HTML Source Code for the Table in Figure 3. Note how ROWSPAN and COLSPAN attributes are used to define cells that span rows and columns respectively, and that the table boxhead (THEAD) and body (TBODY) regions are explicitly labelled. The data source for the table is labelled as a paragraph (P). This type of formatting rather than logical structure description is common in practice, making automated retrieval and clustering tasks for HTML tables difficult[24, 42, 91, 96].

Table I. Available Document Recognition Software

Source	Packages	Web Site	Note
AABBYY	FineReader	http://www.abbyy.com	Commercial
ScanSoft	OmniPage OmniForm TextBridge	http://www.scansoft.com	Commercial
ExperVision	TypeReader WebOCR	http://www.expervision.com	Commercial: WebOCR on-line service is free
CharacTell	Simple OCR	http://www.simpleocr.com	Commercial
Musitek	SmartScore	http://www.musitek.com	Commercial Music Recognition, Scoring
Michael D. Garris (NIST)	Form-Based Handprint Rec. Sys.	http://www.itl.nist.gov/ iaui/894.03/databases/ defs/nist_ocr.html	Free, with source code Unrestricted Use. Large training sets
Donato Malerba et. al.	Wisdom++	http://www.di.uniba.it/ ~malerba/wisdom++/	Free for research and teaching purposes
R. Karpischek et. al.	Clara OCR	http://www.claraocr.org	GPL*
J. Schulenburg et. al.	JOCR	http://jocr.sourceforge.net	GPL*
Klaas Freitag	Kooka	http://www.kde.org/apps/kooka	GPL*, Scanning and OCR interface

*GPL: Freely available under GNU General Public License

Fig. 6. A Table Describing Available Document Recognition Software. This table is from an ASCII text file, and is unruled. Note the title and the footnote below the table; the footnote is referenced using asterisks in the rightmost column of the table. Together, the title and footnote span all the gaps between columns. This type of arrangement complicates the detection of separators in projection profiles (horizontal and vertical histograms of foreground pixels/characters[6,15,28,32,45,50,54,78,86,98]).

and blocks of cells. In Wang's scheme, the table grid and cell topology (cell adjacencies) are defined by a combination of the topology on dimensions and formatting rules.

Wang's model is appealing because it is reasonably simple and separates concerns cleanly, with editing driven by logical structure rather than blocks of cells, as in many conventional table editing tools (e.g. spreadsheets). In Wang's scheme what we have called logical structure is separated into layout, presentation, and logical classes (see Section 4.1 of Wang's thesis[89]). Wang's model does not describe footnotes or other text associated with a table (e.g. titles or captions). Stub heads are assumed to be empty, and headers are assumed to be located only in the boxhead and stub of the table. It is also not designed to handle nested tables, in which cell contents are themselves tables; however, this type of table is fairly unusual.

2.3 Table Models for Recognition

In the literature, table models for recognition must support two tasks: the detection of tables, and the decomposition of table regions into logical structure descriptions. They tend to be more complex than generative models, because they must define and relate additional structures for recovering the components of generative models. Figure 7 presents a number of these additional

Primitive Structures

- Run lengths[15,16,50,54,95]
- Connected components[1,33,34,47,50,54,80,95,98]
- Separators
  - Lines[15,16,46,50,72]
  - Whitespace [22,28,36,41,49,65,86,92,98]
- Intersections
  - Of separators[4,5,15,50,78,84,87,94]
  - Of lines and text[6,34,97]
- Characters
  - Provided in text or markup files[21,24,36,47,65,73,85]
  - From Optical Character Recognition[7,49,57,69,80,86]
- Text lines[46,67,79,80]
- Other Symbols
  - Arrow heads (to repeat cell values[6])
  - X's (to cancel cells[4])

Table-Specific Structures

- Table grid[5,22,28,33,45,54,86,98]
- Cells
  - Multi-line cells[28,38,44,65]
  - Spanning cells[24,69,86]
  - Cell Topology, usually as rows and columns of cells[21,28,33,38,45,65,78,85,98]
- Table regions: boxhead, stub, and body[38,43,73,96]
- Captions, titles, sources, footnotes, and other text associated with tables[20,69,73,86]
- Whole Tables (for table detection[14,24,36,46,48,49,54,65,78,84,91,93])
- Indexing structure
  - Indexing relation for tables[21,24,85,96]
  - Entry structure in tables of contents[7,9,81,82]

Fig. 7. Types of Structures in Table Models for Recognition. For table structure recognition, the most common outputs are the table grid, cell topology, and table regions (body, stub, and boxhead). Less commonly, some papers go further and encode the indexing relation[21,24,85,96] or entry structure in tables of contents[7,9,81,82]. Multi-line cells contain mutliple text lines.

---

structures, such as connected components and line intersections. The usefulness of a table model for recognition in a set of documents is determined by the proportion of tables described by the model, the degree to which the model excludes other types of data (e.g machine drawings), and how reliably the objects and relations of the model can be inferred. The efficacy of a model is difficult to assess in advance of doing a performance evaluation (see Section 6). Usually table models are designed informally by trying to describe the tables in a set of documents (e.g. as was done for Wang's model[89]).

Only a small number of models for recognition have been described explicitly in the literature[21,40,69]. These models are less complete than Wang's. More commonly the structures, relations, and assumptions of a table model for recognition are determined by the sequence of observations, transformations, and inferences used by a table recognizer. As an example, from operations that locate column separators at gaps of a vertical projection, we learn the recognition model has a notion of horizontally adjacent columns, where columns are separated by uninterrupted whitespace gaps; this implicit model cannot describe the table shown in Figure 6. In many papers the description of operations, and thus the reader's view of the table model, is partial.

A table model may be static, where all parameters are fixed before run-time, or adaptive, where parameters are altered at run-time based on input data. Figure 8 lists a number of static and adaptive parameters of table models. Some adaptive parameters used in table recognition are fairly sophisticated, including line grammars for tables of contents[7], and regular-expressions describing the types of cells in a table[69,78]. In the literature, model parameters have been set both manually and using machine learning methods[14,70]. Parameters have included encodings of domain knowledge such a bigrams[41] and ontologies[85] (graph-based knowledge encodings); these encodings are used in the analysis of cell content. Thresholds, tolerances, domain knowledge encodings, and other parameters constrain a table model, and consequently affect what may be inferred by a table recognizer. In this way, they specify a set of assumptions about table location, structure, and content.

As for any other pattern recognition model, there are a number of issues in designing a table model including the 'curse of dimensionality' (the required training sample size growing exponentially as the number of parameters increases) and the complexity of rule-based systems. Perlovsky has provided a brief and informative history of these two problems[68].

3 Observations

Observations measure and collect the data used for decision making in a table recognizer. As shown in Figure 1, observations may be made on any available data; the input document, table model, input parameters, existing hypotheses, or the current set of observations. Figure 8 lists a number of observations made by table recognition systems. Observations may be categorized by the type of data on which an observation is made: images and text files (physical structure), descriptions of table structure (logical structure), sets of existing observations (descriptive statistics), or parameters.

For physical structure, observations include geometry, histograms, and textures. Geometric observations

Physical Structure

- Geometry
  - Height, Width, and Area
  - Aspect ratio (height:width)
  - Angle (e.g. of a line)
  - Skew estimation
    - From skew of detected lines[53]
    - From bounding boxes[46]
  - Docstrum: angle, distance between connected components[67]
  - Overlap of regions (e.g. table regions[14])
  - Region perimeter[34, 78]
  - Representative point
    - Centroid[67]
    - Top-left corner[94]
    - Text baseline y-position[33]
  - Distance between points (e.g. between centroids[67])
- Histograms (Projection Profiles)
  - Projected image pixels[16, 63, 79]
  - Projected bounding boxes[25, 32, 45]
  - Boxes projected as symmetric triangles [98]
  - Boxes projected as 'M' shapes of constant area[49]
  - Weighted projections (e.g. by vertical position[28, 98])
- Texture
  - Value transition count (cross-counts)[50, 86]
  - Pixel density[13]
  - Character density[21]

Parameters

- Static or Adaptive
  - Probability (e.g. for table detection[93])
  - Thresholds (adaptive examples: [32, 46, 98])
  - Tolerances (e.g. for use in X-Y cutting[13])
  - Weights (e.g. for linear combinations[48])
- Adaptive
  - Line grammar (e.g. for table of contents[7])
  - Regular expressions for cell contents[69, 78]
- Encoded Domain Knowledge (Static)
  - Word bigrams[41]
  - Ontologies[85]

Logical Structure

- Table structures (see Figure 7)
- Edit distance[10]
  - Deriving reg. expressions for strings[69, 78]
- Cell block cohesion measures[43, 85, 91]
- Graphs
  - Line intersections[87]
  - Form structure[12]
  - Table indexing structure[22, 39]
- Table Syntax (as grammars; see Figure 12)

Descriptive Statistics

- Cardinality (counting)
- Probability (e.g. computed from a sample)
- Weighted Linear Combinations of Observations
  - 'Columness' of a region[48]
  - 'Tableness' of a region[93]
- Comparisons
  - Difference (e.g. between heights[85])
  - Derivative (e.g. of histograms[16, 79])
  - Inner ('dot') product and cosine of vectors [80, 91, 96]
  - Correlation (e.g. of text line spacings[49])
  - Word uniqueness[91, 96]
- Summary Statistics
  - Range, Mean, Median
  - Variance/standard deviation[46]
  - Periodicity
    - In histograms[79]
    - In column, row structure[91]
    - Line/string periodicity[73]

Fig. 8. Observations in the Table Recognition Literature. Observations are classified based on whether they are taken from an image or text file containing a table (physical structure), from a description of table structure and/or content (logical structure), from a set of existing observations (descriptive statistics), or from system parameters. Static parameters are set before execution; adaptive parameters are set at run-time.

include perimeter, representative points (e.g. centroid), area, height, width, aspect ratio and angle. They also include distances, angles, and areas of overlap between two regions. Histograms are often observed when locating text lines and to define the table grid. Textural features include cross-counts (a transition count for a line of pixels in a binary image or characters in a text file) and density (proportion of 'on' to 'off' pixels in a binary image, or character to blank cells in a text file). Texture metrics have been used to classify regions[50, 86].

For logical structure, observations made include table structures, edit distance, cell cohesion measures, graphs, and table syntax. The edit distance[10] from logical structure description A to logical structure description B is a weighted linear combination of the number of insertions, deletions, and substitutions required to transform A to B. In the table recognition literature, edit distance

Physical Structure

- Image Binarization (e.g.[28,79])
- Image compression
  - Run-length encoding[95]
  - Block adjacency graph[97]
- Image resampling
  - Subsampling[15,34]
  - Supersampling[67]
  - Quadtree[79]
- Hough transform[30] (e.g. for locating lines)
- Affine transformations: rotation, shearing, translation and scaling[77], (e.g. used for de-skewing an image[46])
- Interpolation to recover parts of characters intersected by lines[97]
- Mathematical Morphology[30]
  - RLSA (Run-length smoothing algorithm[95])
  - Dilations and closings
    - In images[66,6]
    - In text files[47]
    - For joining lines[97]
  - Thinning[45]
  - Edge detection[87,94]

Logical Structure

- Merging/splitting of regions
  - Cells[43]
  - Tables[78,96]
  - Splitting region at detected separators[54]
- Graph/tree transformations
  - To correct structural errors[7]
  - Join regions into a table region[76]
- Filtering
  - Small regions for noise reduction[52,66,80]
  - Textures, images and half-tones[80]
- Insertion of table lines[33,54]
- Produce boxes from line intersections [3,87]
- Sorting and Indexing
  - Sorting (e.g. boxes by geometric attributes[8])
  - Indexing (e.g. of cells[22,45])
- Translation
  - HTML to character matrix[24,91]
  - Map strings to regular expressions[69]
  - Transform tokens of a single class to a uniform representation ([65,69])
  - Encoding recognized form data[12,97]
  - Indexing relation of a table[22]

Descriptive Statistics

- Histogram smoothing[79]
- Histogram thresholding

Fig. 9. Transformations in the Table Recognition Literature. Transformations are classified based on whether they are applied to an image or text file containing a table (physical structure), to a description of table structure and/or content (logical structure), or to descriptive statistics.

has been used to derive regular expression 'types' for columns[69,78] and in performance evaluation (see Section 6). Cell cohesion measures[43,85,91] are used to measure whether cells exhibit dimension name and element relationships (e.g. in a column of cells[43,85]), and to determine the consistency of cell properties in a block of cells[91].

For descriptive statistics, observations include cardinality (counting), probabilities, weighted linear combinations of observations, comparisons, and summary statistics. Variance and standard deviation have been employed to define tolerances and thresholds[46]. In addition, periodicity (spatial regularities or intervals) has been used to classify primitive text regions[79] and for detecting regularities in row and column structure (e.g. in HTML[91] and plain text[73]).

Parameters of table recognizers were discussed in Section 2. Here we will elaborate further on the use of domain knowledge observations in table recognition, as this is a promising approach in table recognition. Hurst has provided a number examples in which layout information alone is insufficient for defining table grids, cell scopes, and cell topology[41]; the analysis of table content relative to a domain model is required in these examples. To address this, he and Nasukawa[44] proposed improving cell segmentation by constraining detected text continuations using word bigrams. The constraints afforded by these bigrams appear to improve cell segmentation and topology analysis[41]. In Section 5.2 we describe a system by Tubbs and Embley using the correspondence of cell contents to relationships and concepts in an ontology for detecting header nesting and factoring, and in computing cell cohesion measures[85].

4 Transformations

Transformations restructure existing observations to emphasize features of a data set, to make subsequent observations easier or more reliable. Figure 9 lists transformations used in the table recognition literature. As we did for observations, we classify transformations by the type of data to which they are applied: physical structure, logical structure, or to descriptive statistics.

Physical structure transformations include the Hough Transform[30], which is used to approximate parameters

Decision Tree

- Single Dimension
  - Thresholding (e.g. threshold a 'columness' feature to locate columns[48])
  - Priority of separators (e.g. table lines by thickness[22])
  - Using area to classify noise vs. signal[15,52,66,72]
  - Character class (e.g. alphabetic, non-alphabetic, other[65])
- Multiple Dimensions
  - Connected components
    - Defining[30]
    - Classifying[33,45,46,50]
  - Document region classification[46,50,90]
  - C4.5 decision tree induction[74] (for table detection[65])
  - Word token sets[82]
  - Table/non-table classification[91]
  - Text orientation (vertical vs. horizontal[50])
  - Chain code line segment type[53]

Neural Network

- Optical character recognition[80]
- Logo recognition[12]

Nearest Neighbour

- k-nn (e.g. for defining clusters[67])
- Weighted k-nn (e.g. for table detection[91])

Syntactic

- String matching (e.g. HTML cell types[91])
- Regular expressions (e.g. assigning types to text lines[36,65,86])
- Part of speech tagging (e.g. to classify roles of words in tables of contents[7])

Statistical

- Bayesian Classifier ('Naive Bayes')
  - Table detection[91]
  - Functional class of text block (e.g. author, title for table of contents [81])
- Bayesian network (e.g. assigning labels to regions in tables of contents[81])
- Probabilistic relaxation[77] (assigning labels to words in tables of contents[9])

Fig. 10. Classifiers: Inferences Used to Assign Structure and Relation Types. Classification techniques used in table recognition include decision tree, nearest neighbour, neural network, syntactic, and statistical methods. Decision trees are by far the most common technique.

---

of geometric shapes. The Hough Transform is commonly applied to table images in order to support the detection of table lines. Affine transformations[77] are also commonly used, in particular rotations and shearing transforms are applied to correct rotation and shearing in scanned images[1,66,67].

Other physical structure transformations include the image transformations that are often referred to as 'preprocessing': compression, resampling, binarization, and mathematical morphology. Resampling is used to provide low resolution versions of an input image, as researchers have found that this provides a useful alternate view of the document[34,79]. Mathematical morphology[30] is concerned with set-theoretic operations defining how indexed sets (structuring elements) alter elements of other indexed sets. In the table recognition literature, morphological operations have been applied to both binary images and to text files[47]. Structuring elements used in table recognition are usually horizontal and vertical bars, used to close gaps. These types of structuring elements are used in the run-length smoothing algorithm (RLSA[95]), to thin objects[45], and to detect corners[87].

Logical structure transformations include tree and graph transformations, which have been used to merge and split regions (e.g. into tables[76]) and correct errors in table of contents entries[7]. Other logical structure transformations include filtering small objects assumed to be noise[52,66,80], producing shapes from point lists[3, 87], ordering and indexing objects, and translation to alternate representations (e.g. from HTML to plain text[24, 91]). Green and Krishnamoorthy[22] have provided an elegant translation from recognized table structure to a table's indexing relation using templates.

We quickly note a pair of transformations modifying descriptive statistics. Histogram smoothing[79] and thresholding have been used to reduce variance when trying to locate text lines and separators in projections.

Some of the transformations described in this section produce implicit inferences. As an example, Handley[27] has pointed out that the morphological operations of the Run-Length Smoothing Algorithm[95] concurrently classify regions as foreground or background. Image binarization and noise filtering have the same side-effects. Many systems quietly assume that the regions output by these algorithms are valid foreground regions; a set of hypotheses about foreground regions are immediately accepted as valid.

5 Inferences

5.1 Classifiers, Segmenters, and Parsers

Inferences decide whether or how a table model can be fit to a document, through the generation and testing of hypotheses. More specifically, inferences decide whether physical and logical structures of the table model exist in a document using data observed from the input document, input parameters, table model, transformed observations, and table hypotheses (as shown in Figure 1). As seen in Figures 10, 11, and 12, a large variety of inferencing techniques have be used in table recognition. Comparing inferences, even for the same target structure, is often difficult because different observations or decision techniques are used.

In studying inferences in table recognition, we found the following categorization of techniques to be useful.

- *Classifiers:* assign structure and relation types in the table model to data
- *Segmenters:* determine the existence and scope of a type of table model structure in data
- *Parsers:* produce graphs on structures according to table syntax, defined in the table model

This categorization separates the concepts of typing, locating, and relating structures (determined by classifiers, segmenters, and parsers, respectively). These inferencing classes are interdependent, however. The cyclic dependence between classifiers and segmenters has been well documented[11]. Parsers use classification and segmentation to define inputs for analysis, and in producing a parse (see Figure 12). In the other direction, parse results can be used to provide context in segmentation and classification.

Space does not permit a detailed discussion of the inferencing methods in the literature. Instead we will briefly outline classifiers, segmenters, and parsers used in table recognition as summarized in Figures 10, 11, and 12.

Figure 10 separates classifiers into decision tree, nearest neighbour, neural network, syntactic, and statistical methods. We take a very general view of classification in which assigning any type to data is taken to be classification; this includes identifying an image region as being a connected component, for example. Decision trees are by far the most common classification method used in table recognition. An alternate organization for these classification methods is provided by the types they assign (which correspond to table model structures and relations). This type of organization can be seen in Figure 7, where classes of structures used in table models are listed.

Figure 11 summarizes segmenters, which search data for table model components using a binary classifier. The binary classifier tests the presence or absence of a table model component in a data region, while the objective function of the search controls the scope of segmented regions. As a simple example, consider a connected component segmenter; a simple classification defines connected components, while the objective function of the search insures only the largest connected components are actually segmented. Figure 11 categorizes segmentation in table recognition by whether methods cluster or partition data. Clustering has been used to set parameters adaptively using K-mean clustering[98] and to cluster regions hierarchically based on distance[37]. We include closure on relations as a type of clustering, for example of a proximity relation[47]. Yoshida et. al.[96] have presented a technique for clustering HTML tables in the world wide web, producing meta-tables.

Some partitioning segmentations are very simple, such as when rows are segmented at blank lines in text files[47, 65]. Others are more sophisticated, such as segmenters used in table detection. An important differentiating feature in table detection methods is the type of search used. These have included best first search[93], dynamic programming[36], the simplex algorithm[14], and Cesarini et. al.'s iterative decreasing step[14]. The most commonly used recursive partitioning methods in table recognition are variants of the X-Y cut algorithm of Nagy and Seth[63].

Clustering

- Connected components
  - Creation (e.g. for adjacent pixels[30], for adjacent word boxes[47])
  - Clustering connected components[46]
- Tables by content[96]
- K-means clustering (of projection histogram groups[98])
- Hierarchical clustering of regions by distance[36]
- Transitive closure (e.g. of a proximity relation [47])

Partitioning

- Using breadth first search (e.g. to segment columns[37])
- Using best-first search (e.g. to recover body, stub, and boxhead[43])
- Table detection
  - Using dynamic programming[36]
  - Using best-first search[93]
  - Using simplex[64] algorithm[14]
  - Using iterative decreasing step[14]
- Recursive Partitioning
  - X-Y cut[63]: alternating horizontal and vertical partitions at projection histogram minima[22]
  - Modified X-Y cuts, using histogram minima and lines[13]
  - Recursive line partitioning (e.g. by 'best' separating line[50], by line separator priority[22])
- Exact Matching
  - Splitting text columns into rows at blank lines[47, 65]

Fig. 11. Segmenters: Inferences Used to Locate Structures. Segmenters employ a binary classifier and a search function to locate table model components. Target regions matched by the classifier that also satisfy the objective function of the search are clustered or partitioned within the data set.

Hidden Markov Models

Maximizing region adjacency probabilities[90]

Attributed Context-Free Grammars

Tables in images (with parse control-flow directives: [18])
Table form box structure[3, 8, 94]
Form structure[19]
Using input tokens with multiple types to parse tables of contents[82]
For tables in HTML files[91]
For page segmentation[52, 88]

Graph Grammars

Table form cell composition[2]
Table structure from word boxes[76]

Fig. 12. Parsers: Inferences Used to Relate Structures. Parsers produce graphs describing the logical structure of table model components. Here parsing techniques are categorized by the type of grammar encoding logical structure syntax: Hidden Markov models, attributed context-free grammars, and graph grammars. In the process of defining relational structure, parsers both segment and classify data. Consider the simple production rule $A \rightarrow BC$. Applying this rule in a parse clusters type 'B' and 'C' elements in the specified order to produce a type 'A' object.

Recursive partitioning methods actually produce a parse as well as segment regions, as the result describes a hierarchy of regions which may be used to determine the table grid and cell topology, for example[22].

Parsers used in the literature are summarized in Figure 12. Parsers produce graphs describing logical structure according to table syntax defined in a grammar. The grammars used in parsing are part of the table model. Parsers have been used to apply probabilistic constraints on region types[90], for table detection[19, 76], to define page grammars for X-Y cut-based analysis[52, 88], and to parse entry structure in tables of contents[82]. The parsing technique proposed by Takasu et. al.[82] is interesting, because input tokens are initially provided with sets of possible types assigned by a decision tree which the parser then constrains to produce a valid interpretation. In the final interpretation, tokens are assigned a single type.

A small number of techniques for automatically inducing grammars from data have been described in the literature. Takasu et. al.[83] have described a method for inducing a grammar for tables of contents from labelled data. Adaptive methods have been used to define regular expressions describing the data types of cell contents[69, 78], and a context-free grammar of table of contents entry structure[7].

5.2 Inference Sequencing

Table recognizers are frequently required to make inferences based on hypotheses produced by other inferences. For example, many systems will generate hypotheses of line locations. Inferring the location of line intersections needs to assume temporarily that these line hypotheses are valid. At a later point these line hypotheses may be found to be invalid by another inference. In many cases however, once hypotheses are accepted, they are not reconsidered. Rus and Subramanian have proposed an elegant graph-based notation for describing this type of architecture without feedback[78].

Tubbs and Embley[85] have proposed an alternate architecture for recognizing genealogical tables, making use of an ontology describing inheritance. The role of the ontology may be understood as a parameter of their table model (see Section 3). Input to the system describes cell locations and their text contents. Potentially valid hypotheses regarding cell topology, entry structure, and properties of cell contents defined using the ontology (e.g. dimension name and element relationships) are placed in matrices. An iterative control structure is then used to alter confidence weights associated with each hypothesis using an ordered rule set. Iteration stops when a fixed point is reached for hypothesis weights, or after a fixed number of iterations. Decision rules then determine which hypotheses to accept, and how these accepted hypotheses are integrated into an interpretation of entry structure. In this scheme hypotheses dynamically affect one another through feedback.

Commonly in the literature an ordered sequence of classifiers and segmenters is used to infer instances of table model structures. As an example, consider a method in which table cells are segmented, and then columns of cells are segmented. This process can be considered a simple form of parsing, in which a hierarchy defining the composition of objects is defined. For the previous example, the composition of the column is described by the set of segmented cells. X-Y cutting[63] is another example, where the recursive segmentation of regions produces a tree.

One advantage of explicit table models is that their syntax may be encoded in a grammar. How the grammar is applied for table detection or structure recognition can then be controlled using different parsing algorithms, which result in different operation sequences. This ability to specify search strategy as a table recognizer parameter is useful both for characterizing and comparing methods.

6 Performance Evaluation

After the design of a table recognizer is complete, the designer will have some questions. How fast is the system? How well does the table model fit tables in the intended domain (e.g. images of technical articles, or HTML pages on the world wide web)? How reliable are the inferences made by the system, or put differently, what are the type and frequency of errors made by the system? In this section we describe the methods and metrics used in performance evaluation of table recognition, which are then used to address the last two questions.

In order to train or evaluate a table recognizer the logical and/or physical structure of tables in documents must be encoded; this encoding is referred to as ground truth. It is produced using another table model, which must be able to describe the outputs of the table recognizer[35]. Ground truth may be produced manually with interactive tools[39,75] or by automatically generating document sets with ground truth[56,71,92]. Automatic generation permits a degree of experimental control not possible with a random document sample, but then correspondence to real document sets must be addressed. Whether real or generated automatically, many tables have ambiguities, permitting mutliple interpretations[36,39]. An open problem is how, or whether, to encode ground truth for these documents[35,57].

In the table recognition literature three methods have been used to separate available documents with ground truth into training and testing sets: resubstitution, in which all documents are used in training and testing; the 'circuit-training' or leave-one-out' method, in which each document is tested exactly once, with remaining documents used for training each time, or by randomly assigning documents to the training and testing sets. Resubstitution is seldom used as it produces positively biased results[30].

Comparing systems in the literature is difficult, because table models are usually not explicitly defined and differ significantly between systems, there are no available benchmark data sets with ground truth, and systems are seldom described in enough detail to permit replication. However, one trend appears to be that table recognizers using detailed, narrowly-defined models to recover tables in well-characterized document sets appear to perform their intended tasks best (for a good example of this, see Shamillian et. al.[80]). This is explained in part by substantial a priori knowledge permitting strong, well-grounded assumptions to be incorporated into the table model. This reduces the number of table model parameters necessary, making both training and understanding the behaviour of the table recognizer simpler.

In the remainder of this section we describe the performance metrics used in table detection and structure recognition, and mention some experimental design issues.

6.1 Recall and Precision

For classification, given a class X and a set of input patterns with associated ground-truth, recall is the percentage of type X input patterns classified correctly, and precision is the percentage of input patterns assigned type X correctly. Over the set of all possible classes, recall and precision are equivalent. For individual classes, the number of correct class X classifications need not match the number of items correctly assigned class X. To see this, consider the case where a binary classifier (returning X or Y) always assigns class X.

For evaluating the segmentation of regions (e.g. for table detection), the splitting and merging of ground truth regions needs to be taken into account[39,56]. To accommodate this, modified recall and precision metrics that make use of the area of overlap between detected and ground truth regions have been devised[14,48]. The area of overlap between regions in recognizer output and regions in ground truth are then used as weights in modified recall and precision metrics that take merging and splitting into account.

Precision and recall are sometimes combined into a single 'F-measure.' In the table recognition literature this is non-standard, as both the arithmetic and harmonic mean of precision and recall have been called an 'F-measure'[24,85]. The harmonic mean of precision (P) and recall (R) is defined as:

$$H(R,P) = \frac{2RP}{R+P}$$

The two metrics have different biases. For example, given the sum of recall and precision $T = R+P$, the maximum harmonic mean value is obtained when $R = P = T/2$. In contrast, for the arithmetic mean the resulting value depends only on the sum of recall and precision, and not on their relative sizes.

6.2 Edit Distance and Graph Probing

Edit distance (see Section 3) has been used to compare detected tables to ground-truth (taking false positives, false negatives, merges, and splits into account[36]). It has also been used in table structure recognition to compare graphs describing the logical structure of tables[60]. Edit distance has some associated problems: the minimal editing sequence is not necessarily unique, it can be computationally intensive, and appropriately setting the weights of operations is difficult. In the literature the operation weights are always set to one.

To address these problems a new metric called table agreement has been proposed[39]. Agreement is computed by automatically generating queries about the number, content, topology, and indexing structure of cells in logical table structure encodings for recognizer output and ground truth. Queries are verified or contradicted by searching the other table encoding; agreement is defined as the percentage of verified queries. This process of generating and verifying queries from graphs is called graph probing[60].

6.3 Experimental Design

Evaluation in table recognition is still maturing. Performance evaluations are usually made from final outputs, ignoring individual inferences. As a result, the effects of individual decisions are often confounded (inseparable) in the evaluation. Wang and Hu have made a step forward in this regard. They performed an experiment making use of a fixed classification method (an induced decision tree) while varying the set of observations used for training the classifier[91]. In their design, the inference method and observations are clearly separated into independent factors.

A number of other experimental design issues remain in the area. These include a need for better sampling[57] and comparisons of experimental conditions[61]. A test worth considering for the robust comparison of conditions is the Analysis of Variance (ANOVA[31]).

7 Conclusion

We have presented the table recognition literature from the viewpoint that table recognizers may be understood as sequences of decisions (inferences) supported by observations and transformations of available data. These decisions are made relative to a table model describing the location and composition of tables in a set of documents. Figure 7 summarizes the structures used in table models for recognition. Observations, transformations, and inferences made in the table recognition literature are summarized in Figures 8, 9, 10, 11, and 12. In Section 6 we describe performance evaluation methods for determining the sufficiency of a table recognizer for recovering and/or analyzing tables in a set of documents. We have tried in our discussion to point out various assumptions inherent or implied in different operations and sequences of operations in the literature.

As pointed out in Section 6, it appears at present that simple, domain-specific table models have more promise than complex, general models. It may be worth studying whether combinations of table recognizers with simple models yield improvements in performance over the current state-of-the-art for large, heterogeneous document sets such as technical articles.

Other avenues for future work include extending the proposed table models of Wang[89] and others, further exploring content-based methods for recognition (including cell cohesion metrics and the use of domain knowledge), improving experimental design and evaluation techniques at both the level of individual decisions and whole systems, defining corpii of ground-truth tables for use in the community, and exploring new observations, transformations, and inferences for use in table recognition.

Acknowledgements. The authors wish to thank George Nagy for providing references and helpful discussion, Bob Tennent for his comments on document semantics, and the anonymous referees for suggestions which aided in substantially improving this paper. This research was supported by the Natural Sciences and Engineering Research Council of Canada.

References

1. A.A. Abu-Tarif. Table processing and understanding. Master's thesis, Rensselaer Polytechnic Institute, 1998.
2. A. Amano and N. Asada. Complex table form analysis using graph grammar. Lecture Notes in Computer Science, vol. 2423, pp. 283–386. Springer-Verlag, Berlin, 2002.
3. A. Amano, N. Asada, T. Motoyama, T. Sumiyoshi, and K. Suzuki. Table form document synthesis by grammar-based structure analysis. Proc. Sixth Int'l Conf. Document Analysis and Recognition, pp. 533–537, Seattle, WA, 2001.
4. J.F. Arias, A. Chhabra, and V. Misra. Efficient interpretation of tabular documents. Proc. Thirteenth Int'l Conf. Pattern Recognition, pp. 681–685, Vienna, Austria, 1996.
5. J.F. Arias, A. Chhabra, and V. Misra. Interpreting and representing tabular documents. Proc. Conf. Computer Vision and Pattern Recognition, pp. 600–605, San Francisco, CA, 1996.
6. S. Balasubramanian, S. Chandran, J. Arias, and R. Kasturi. Information extraction from tabular drawings. Proc. Document Recognition I (IS&T/SPIE Electronic Imaging), vol. 2181, pp. 152–163, San Jose, CA, 1994.
7. A. Belaïd. Recognition of table of contents for electronic library consulting. Int'l J. Document Analysis and Recognition, 4(1):35–45, 2001.
8. L. Bing, J. Zao, and X. Hong. New method for logical structure extraction of form document image. Proc. Document Recognition and Retrieval VI (IS&T/SPIE Electronic Imaging), vol. 3651, pp. 183–193, San Jose, CA, 1999.
9. F.L. Bourgeois, H. Emptoz, and S.S. Bensafi. Document understanding using probabilistic relaxation: Application on tables of contents of periodicals. Proc. Sixth Int'l Conf. Document Analysis and Recognition, pp. 508–512, Seattle, WA, 2001.
10. H. Bunke. Structural and syntactic pattern recognition. C. H. Chen, L. F. Pau, and P.S.P. Wang, editors, Handbook of Pattern Recognition and Computer Vision, pp. 163–209. World Scientific, Singapore, 1993.
11. R.G. Casey and E. Lecolinet. A survey of methods and strategies in character segmentation. IEEE Trans. Pattern Analysis and Machine Intelligence, 18(7):690–706, 1996.
12. F. Cesarini, M. Gori, S. Marinai, and G. Soda. INFORMys: A flexible invoice-like form-reader system. IEEE Trans. Pattern Analysis and Machine Intelligence, 20(7):730–745, 1998.
13. F. Cesarini, M. Gori, S. Marinai, and G. Soda. Structured document segmentation and representation by the modified X-Y tree. Proc. Fifth Int'l Conf. Document Analysis and Recognition, pp. 563–566, Bangalore, India, 1999.
14. F. Cesarini, S. Marinai, L. Sarti, and G. Soga. Trainable table location in document images. Proc. Sixteenth Int'l Conf. Pattern Recognition, vol. 3, pp. 236–240, Québec City, Canada, 2002.
15. S. Chandran and R. Kasturi. Structural recognition of tabulated data. Proc. Second Int'l Conf. Document Analysis and Recognition, pp. 516–519, Tsukuba Science City, Japan, 1993.
16. A.K. Chhabra, V. Misra, and J. Arias. Detection of horizontal lines in noisy run length encoded images: the FAST method. Lecture Notes in Computer Science, vol. 1072, pp. 35–48. Springer-Verlag, Berlin, 1996.
17. R.A. Coll, J.H. Coll, and G. Thakur. Graphs and tables: a four-factor experiment. Comm. ACM, 37(4):76–86, 1994.
18. B. Coüasnon. DMOS: A generic document recognition method, application to an automatic generator of musical scores, mathematical formulae and table recognition

systems. Proc. Sixth Int'l Conf. Document Analysis and Recognition, pp. 215–220, Seattle, WA, 2001.
19. B. Coüasnon and L. Pasquer. A real-world evaluation of a generic document recognition method applied to a military form of the 19th century. Proc. Sixth Int'l Conf. Document Analysis and Recognition, pp. 779–783, Seattle, WA, 2001.
20. S. Douglas and M. Hurst. Layout and language: Lists and tables in technical documents. Proc. ACL SIGPARSE Workshop on Punctuation in Computational Linguistics, pp. 19–24, Santa Cruz, CA, 1996.
21. S. Douglas, M. Hurst, and D. Quinn. Using natural language processing for identifying and interpreting tables in plain text. Proc. Fourth Ann. Symp. Document Analysis and Information Retrieval, pp. 535–546, Las Vegas, NV, 1995.
22. E. Green and M. Krishnamoorthy. Model-based analysis of printed tables. Lecture Notes in Computer Science, vol. 1072, pp. 80–91. Springer-Verlag, Berlin, 1996.
23. J. Grossman, editor. Chicago Manual of Style, chapter 12 (Tables). University of Chicago Press, 14th edition, 1993.
24. S. Tsai H. Chen and J. Tsai. Mining tables from large scale HTML texts. Proc. Eighteenth Int'l Conf. Computational Linguistics, Saarbrucken, Germany, 2000.
25. J. Ha, R.M. Haralick, and I.T. Phillips. Recursive X-Y cut using bounding boxes of connected components. Proc. Third Int'l Conf. Document Analysis and Recognition, pp. 952–955, Montreal, Canada, 1995.
26. R. Hall. Handbook of Tabular Presentation. The Ronald Press Company, New York, 1943.
27. J.C. Handley. Electronic Imaging Technology, chapter 8 (Document Recognition). IS&T/SPIE Optical Engineering Press, Bellingham, WA, 1999.
28. J.C. Handley. Table analysis for multi-line cell identification. Proc. Document Recognition and Retrieval VIII (IS&T/SPIE Electronic Imaging), vol. 4307, pp. 34–43, San Jose, CA, 2001.
29. R.M. Haralick. Document image understanding: Geometric and logical layout. Proc. Conf. Computer Vision and Pattern Recognition, pp. 385–390, Seattle, WA, 1994.
30. R.M. Haralick and L.G. Shapiro. Computer and Robot Vision (2 vols). Addison-Wesley, Reading, MA, 1992.
31. K. Hinkelmann and O. Kempthorne. Design and Analysis of Experiments: Introduction to Experimental Design, vol. 1. John Wiley and Sons Inc., New York, 1994.
32. Y. Hirayama. A block segmentation method for document images with complicated column structures. Proc. Second Int'l Conf. Document Analysis and Recognition, pp. 91–94, Tsukuba Science City, Japan, 1993.
33. Y. Hirayama. A method for table structure analysis using DP matching. Proc. Third Int'l Conf. Document Analysis and Recognition, pp. 583–586, Montreal, Canada, 1995.
34. O. Hori and D.S. Doermann. Robust table-form structure analysis based on box-driven reasoning. Proc. Third Int'l Conf. Document Analysis and Recognition, pp. 218–221, Montreal, Canada, 1995.
35. J. Hu, R. Kashi, D. Lopresti, G. Nagy, and G. Wilfong. Why table ground-truthing is hard. Proc. Sixth Int'l Conf. Document Analysis and Recognition, pp. 129–133, Seattle, WA, 2001.
36. J. Hu, R. Kashi, D. Lopresti, and G. Wilfong. Medium-independent table detection. Proc. Document Recognition and Retrieval VII (IS&T/SPIE Electronic Imaging), vol. 3967, pp. 291–302, San Jose, CA, 2000.
37. J. Hu, R. Kashi, D. Lopresti, and G. Wilfong. Experiments in table recognition. Proc. Workshop on Document Layout Interpretation and Applications, Seattle, WA, 2001.
38. J. Hu, R. Kashi, D. Lopresti, and G. Wilfong. Table structure recognition and its evaluation. Proc. Document Recognition and Retrieval VIII (IS&T/SPIE Electronic Imaging), vol. 4307, pp. 44–55, San Jose, CA, 2001.
39. J. Hu, R.S. Kashi, D. Lopresti, and G.T. Wilfong. Evaluating the performance of table processing algorithms. Int'l J. Document Analysis and Recognition, 4(3):140–153, 2002.
40. M. Hurst. Layout and language: Beyond simple text for information interaction - modelling the table. Proc. Second Int'l Conf. Multimodal Interfaces, Hong Kong, 1999.
41. M. Hurst. Layout and language: An efficient algorithm for detecting text blocks based on spatial and linguistic evidence. Proc. Document Recognition and Retrieval VIII (IS&T/SPIE Electronic Imaging), vol. 4307, pp. 56–67, San Jose, CA, 2001.
42. M. Hurst. Layout and language: Challenges for table understanding on the web. Proc. First Int'l Workshop on Web Document Analysis, pp. 27–30, Seattle, WA, 2001.
43. M. Hurst and S. Douglas. Layout and language: Preliminary investigations in recognizing the structure of tables. Proc. Fourth Int'l Conf. Document Analysis and Recognition, pp. 1043–1047, Ulm, Germany, 1997.
44. M. Hurst and T. Nasukawa. Layout and language: Integrating spatial and linguistic knowledge for layout understanding tasks. Proc. Eighteenth Int'l Conf. Computational Linguistics, Saarbrucken, Germany, 2000.
45. K. Itonori. Table structure recognition based on textblock arrangement and ruled line position. Proc. Second Int'l Conf. Document Analysis and Recognition, pp. 765–768, Tsukuba Science City, Japan, 1993.
46. A.K. Jain and B. Yu. Document representation and its application to page decomposition. IEEE Trans. Pattern Analysis and Machine Intelligence, 20(3):294–308, 1998.
47. T.G. Kieninger. Table structure recognition based on robust block segmentation. Proc. Document Recognition V (IS&T/SPIE Electronic Imaging), vol. 3305, pp. 22–32, San Jose, CA, 1998.
48. T.G. Kieninger and A. Dengel. Applying the T-RECS table recognition system to the business letter domain. Proc. Sixth Int'l Conf. Document Analysis and Recognition, pp. 518–522, Seattle, WA, 2001.
49. B. Klein, S. Gökkus, T. Kieninger, and A. Dengel. Three approaches to "industrial" table spotting. Proc. Sixth Int'l Conf. Document Analysis and Recognition, pp. 513–517, Seattle, WA, 2001.
50. H. Kojima and T. Akiyama. Table recognition for automated document entry system. High-Speed Inspection Architectures, Barcoding, and Character Recognition (Proc. SPIE), vol. 1384, pp. 285–292, Boston, MA, 1990.
51. W. Kornfeld and J. Wattecamps. Automatically locating, extracting and analyzing tabular data. Proc. Twenty-first Int'l ACM SIGIR Conf. Research and Development in Information Retrieval, pp. 347–348, Melbourne, Australia, 1998.

52. S. Krishnamoorthy, G. Nagy, S. Seth, and M. Viswanathan. Syntactic segmentation and labelling of digitized pages from technical journals. IEEE Trans. Pattern Analysis and Machine Intelligence, 15(7):737–747, 1993.
53. S.W. Lam, L. Javanbakht, and S.N. Srihari. Anatomy of a form reader. Proc. Second Int'l Conf. Document Analysis and Recognition, pp. 506–509, Tsukuba Science City, Japan, 1993.
54. A. Laurentini and P. Viada. Identifying and understanding tabular material in compound documents. Proc. Eleventh Int'l Conf. Pattern Recognition, pp. 405–409, The Hague, Netherlands, 1992.
55. S. Lewandowksy and I. Spence. The perception of statistical graphs. Sociological Methods and Research, 18(2 & 3):200–242, 1989.
56. J. Liang. Document Structure Analysis and Performance Evaluation. PhD thesis, University of WA, 1999.
57. D. Lopresti. Exploiting WWW resources in experimental document analysis research. Lecture Notes in Computer Science, vol. 2423, pp. 532–543. Springer-Verlag, Berlin, 2002.
58. D. Lopresti and G. Nagy. Automated table processing: An (opinionated) survey. Proc. Third Int'l Workshop on Graphics Recognition, pp. 109–134, Jaipur, India, 1999.
59. D. Lopresti and G. Nagy. A tabular survey of automated table processing. Lecture Notes in Computer Science, vol. 1941, pp. 93–120. Springer-Verlag, Berlin, 2000.
60. D. Lopresti and G. Wilfong. Evaluating document analysis results via graph probing. Proc. Sixth Int'l Conf. Document Analysis and Recognition, pp. 116–120, Seattle, WA, 2001.
61. S. Mao and T. Kanungo. Empirical performance evaluation methodology and its application to page segmentation algorithms. IEEE Trans. Pattern Analysis and Machine Intelligence, 23(3):242–256, 2001.
62. G. Nagy. Twenty years of document image analysis in PAMI. IEEE Trans. Pattern Analysis and Machine Intelligence, 22(1):38–62, 2000.
63. G. Nagy and S. Seth. Hierarchical representation of optically scanned documents. Proc. Seventh Int'l Conf. Pattern Recognition, pp. 347–349, Montreal, Canada, 1984.
64. J.A. Nelder and R. Mead. A simplex method for function minimization. Computer Journal, (7):308–313, 1965.
65. H.T. Ng, C.Y. Lim, and J.L.T. Koo. Learning to recognize tables in free text. Proc. Thirty-Seventh Ann. Meet. Assn. Computational Linguistics, pp. 443–450, College Park, MD, 1999.
66. L. O'Gorman. Image and document processing techniques for the RightPages electronic library system. Proc. Eleventh Int'l Conf. Pattern Recognition, pp. 260–263, The Hague, Netherlands, 1992.
67. L. O'Gorman. The document spectrum for page layout analysis. IEEE Trans. Pattern Analysis and Machine Intelligence, 15(11):1162–1173, 1993.
68. L.I. Perlovsky. Conundrum of combinatorial complexity. IEEE Trans. Pattern Analysis and Machine Intelligence, 20(6):666–670, 1998.
69. C. Peterman, C.H. Chang, and H. Alam. A system for table understanding. Proc. Document Image Understanding Technology, pp. 55–62, Annapolis, MD, 1997.
70. M. Petrou. Learning in pattern recognition. Lecture Notes in Computer Science, vol. 1715, pp. 1–12. Springer-Verlag, Berlin, 1999.
71. I. Phillips and A.K. Chhabra. Empirical performance evaluation of graphics recognition systems. IEEE Trans. Pattern Analysis and Machine Intelligence, 21(9):849–870, 1999.
72. A. Pizano. Extracting line features from images of business forms and tables. Proc. Eleventh Int'l Conf. Pattern Recognition, pp. 399–403, The Hague, Netherlands, 1992.
73. P. Pyreddy and W.B. Croft. Tintin: A system for retrieval in text tables. Proc. Second Int'l Conf. Digital Libraries, pp. 193–200, Austin, TX, 1997.
74. J.R. Quinlan. C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco, CA, 1993.
75. RAF Technology, Redmond, WA. Illuminator User's Manual, 1995.
76. M.A. Rahgozar and R. Cooperman. A graph-based table recognition system. Proc. Document Recognition III (IS&T/SPIE Electronic Imaging), vol. 2660, pp. 192–203, San Jose, CA, 1996.
77. A. Rosenfeld and A.C. Kak. Digital Picture Processing (2 vols). Academic Press, Orlando, FL, 1982.
78. D. Rus and D. Subramanian. Customizing information capture and access. ACM Trans. Information Systems, 15(1):67–101, 1997.
79. L. Seong-Whan and R. Dae-Seok. Parameter-free geometric document layout analysis. IEEE Trans. Pattern Analysis and Machine Intelligence, 23(11):1240–1256, 2001.
80. J.H. Shamillian, H.S. Baird, and T.L. Wood. A retargetable table reader. Proc. Fourth Int'l Conf. Document Analysis and Recognition, pp. 158–163, Ulm, Germany, 1997.
81. S. Souafi-Bensafi, M. Parizeau, F. Lebourgeois, and H. Emptoz. Bayesian networks classifiers applied to documents. Proc. Sixth Int'l Conf. Document Analysis and Recognition, pp. 508–511, Seattle, WA, 2001.
82. A. Takasu, S. Satoh, and E. Katsura. A document understanding method for database construction of an electronic library. Proc. Twelfth Int'l Conf. Pattern Recognition, pp. 463–466, Jerusalem, Israel, 1994.
83. A. Takasu, S. Satoh, and E. Katsura. A rule learning method for academic document image processing. Proc. Third Int'l Conf. Document Analysis and Recognition, pp. 239–242, Montreal, Canada, 1995.
84. W. Tersteegen and C. Wenzel. Scantab: Table recognition by reference tables. Proc. Third Workshop on Document Analysis Systems, Nagano, Japan, 1998.
85. K.M. Tubbs and D.W. Embley. Recognizing records from the extracted cells of microfilm tables. Proc. ACM Symp. Document Engineering, pp. 149–156, McLean, VA, 2002.
86. S. Tupaj, Z. Shi, C.H. Chang, and H. Alam. Extracting tabular information from text files. Available from citeseer: http://citeseer.nj.nec.com/tupaj96extracting.html, 1996.
87. E. Turolla, A. Belaid, and A. Belaid. Form item extraction based on line searching. Lecture Notes in Computer Science, vol. 1072, pp. 69–79. Springer-Verlag, Berlin, 1996.
88. M. Viswanathan, E. Green, and M.S. Krishnamoorthy. Document recognition: An attribute grammar approach. Proc. Document Recognition III (IS&T/SPIE Electronic Imaging), vol. 2660, pp. 101–111, San Jose, CA, 1996.
89. X. Wang. Tabular Abstraction, Editing and Formatting. PhD thesis, University of Waterloo, Canada, 1996.

90. Y. Wang, R. Haralick, and I.T. Phillips. Zone content classification and its performance evaluation. Proc. Sixth Int'l Conf. Document Analysis and Recognition, pp. 540–544, Seattle, WA, 2001.
91. Y. Wang and J. Hu. Detecting tables in HTML documents. Lecture Notes in Computer Science, vol. 2423, pp. 249–260. Springer-Verlag, Berlin, 2002.
92. Y. Wang, I.T. Phillips, and R. Haralick. Automatic table ground truth generation and a background-analysis-based table structure extraction method. Proc. Sixth Int'l Conf. Document Analysis and Recognition, pp. 528–532, Seattle, WA, 2001.
93. Y. Wang, T. Phillips, and R.M. Haralick. Table detection via probability optimization. Lecture Notes in Computer Science, vol. 2423, pp. 272–282. Springer-Verlag, Berlin, 2002.
94. T. Watanabe, Q. Luo, and N. Sugie. Layout recognition of multi-kinds of table-form documents. IEEE Trans. Pattern Analysis and Machine Intelligence, 17(4):432–445, 1995.
95. K.Y. Wong, R.G. Casey, and F.M. Wahl. Document analysis system. IBM Journal of Research and Development, 26(6):647–656, 1982.
96. M. Yoshida, K. Torisawa, and J. Tsujii. A method to integrate tables of the world wide web. Proc. First Int'l Workshop on Web Document Analysis, pp. 31–34, Seattle, WA, 2001.
97. B. Yu and A.K. Jain. A generic system for form dropout. IEEE Trans. Pattern Analysis and Machine Intelligence, 18(11):1127–1134, 1996.
98. K. Zuyev. Table image segmentation. Proc. Fourth Int'l Conf. Document Analysis and Recognition, pp. 705–708, Ulm, Germany, 1997.

Richard Zanibbi is currently completing a Ph.D. in computer science at Queen's University, Canada. From Queen's University he has received an M.Sc. in computer science (2000), a B.A. (1998), and a Bachelor of Music (1998). In addition to his studies, Richard has worked as a research programmer for Legasys Corporation (1999-2000), as a research assistant in the Medical Computing Lab at Queen's University (2001-2003), and as a summer intern for Xerox Corporation (2003). His research interests include document recognition, pattern recognition, medical computing, and human-computer interaction.

Dorothea Blostein received an M.Sc. degree in computer science from Carnegie Mellon University in 1980, and a Ph.D. from the University of Illinois in 1987. Since 1988 she has been a professor in the School of Computing, Queen's University, Kingston, Ontario. Dr. Blostein was chair of GREC 2001, the Fourth IAPR International Workshop on Graphics Recognition. She has served on a number of program committees, including Diagrams 2004 and 2002, GREC 2003 and 1999, ICPR 2002, WEDELMUSIC 2001, and AGTIVE'99.

Dr. Blostein's research interests include pattern recognition, document image analysis and graph transformation. A particular interest is the relationship between generation and recognition of diagrams. She is co-author of Lime, a sophisticated editor for music notation.

James R. Cordy is Professor and Director of the School of Computing and Professor of Electrical & Computer Engineering at Queen's University, Kingston, Ontario, Canada. From 1995 to 2000 he was vice president and chief research scientist at Legasys Corporation, a software technology company specializing in legacy software system analysis and renovation.

Professor Cordy is the author or co-author of numerous contributions in computer software systems, including the PL/I subset compiler SP/k (1977), the Toronto Euclid compiler (1980), the S/SL compiler technology (1980), the Concurrent Euclid programming language (1981), the Turing programming language (1983), the orthogonal code generation compiler technology (1986), the TXL programming language (1991), the LS/2000 year 2000 conversion system (1996), and the LS/AMT software analysis and migration system (1999).

Dr. Cordy is a member of the ACM and the IEEE.