# THE T-RECS APPROACH FOR TABLE STRUCTURE RECOGNITION AND TABLE BORDER DETERMINATION

Thomas G. Kieninger and Andreas Dengel
German Research Center for Artificial Intelligence (DFKI GmbH)
P.O.Box 2080, 67608 Kaiserslautern, Germany
Phone: +49-631-205-3485, Fax: +49-631-205-3210
E-mail: kieninger@dfki.de
WWW: http://www.dfki.uni-kl.de/~kieni/

**Abstract**

We present a snapshot of the ongoing research in the field of table structure recognition and analysis. The prototypical *T-Recs* system (Table RECognition System) relies on the word level layout (bounding box geometry) as primary input. It moreover considers the textual information and potentially available delineations as further input.

This article resumes the basic ideas and system features as described in [1] and [2] (downloadable from our demo page: `www.dfki.uni-kl.de/~kieni/t_recs/`). This page also allows to interactivly load and change some predefined documents which demonstrate the main features and strengths of the approach.

Next, we will sketch some problems which encounter when applying *T-Recs* to business letters like offers, invoices etc. and discuss appropriate solutions for that problem.

## 1 SUMMARY OF CURRENT SYSTEM

Based on the word level layout information which can be derived from either OCR documents (e.g. the Xerox XDOC-format) or plain ASCII files by using a built-in preprocessor, the *T-Recs* system performs a direct bottom-up clustering of word segments to blocks (not the conventional word-line-block order!) and thus transforms the document representation from the word level layout to a partial logical representation (see Figure 1).

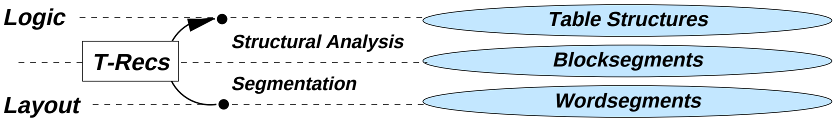

Figure 1: The *T-Recs* contribution to document analysis.

### 1.1 The *T-Recs* Segmentation

The initial block clustering is based on the so called *horizontal-overlap* relation $ovl(w_1, w_2)$. This relation is given for two words in directly adjacent lines if they have a common x-range. Starting with an arbitrary word as the so called *block seed* a new block is constructed as the transitive hull $ovl^*()$ of that relation.

The effect of this clustering is visualized for a small sample block which is seen in Figure 2 (left) using the word “consists” as the seed. The gray stripe (center) indicates the range

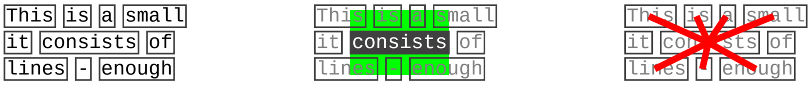

Figure 2: Vertical neighbors of the word "consists".

in which overlapping words might be found. The *segmentation graph* (right) indicates the words that overlap with the seed.

It is obvious that this approach is able to detect even smallest column gaps since the words of adjacent columns do not overlap mutually. But this approach also points out some weakpoints:

- First, it fails if two or more columns have a common header which "glues" the columns together or if the table (and its columns) are directly located next to a regular block with no vertical space.
- The second error is caused by the so called *rivers of white space* that are caused by an occasional space at the same x-position throughout the whole block. The system would not find any linking element between left and right part and hence would not be able to detect this block as one unit.
- The last error class is caused by isolated lines such as headers or words that stick out of the end of a block. In this case there will not be any linking element and hence these words would each build its own block. Merging of these words to left and right neighbors has to be done with caution in order not to loose the narrow column gaps.

We provide a series of postprocessing steps which are able to identify and correct all these missegmentations. Detailled descriptions of these processing steps can be found in [1].

At this point, all table cells only appear as aggregations of columns and not as individual textual units. To achieve a homogeneous view to all basic textual units, we provide a twostep column decomposition approach.

### 1.2 The *T-Recs* Layout Analysis

After this goal directed block segmentation the system continues with the layout analysis. This is done by determining the proper rows and columns and by assigning the data cells to them. Our approach starts determining the proper column and row separators and builds a regular two-dimensional structure which we call the *tiles*. One tile can be covered by at most one block. At the other hand, one block can be covered by several tiles and some tiles can remain empty. Based on these different states we are able to provide simple output routines for some structured format, e.g. HTML. A complete description of these processing steps can be found in [2].

### 1.3 Major System Features

The *T-Recs* system can be characterized by the following features:

- *No delineation* and *no significantly large white spaces* required
- *No accurate column alignment* of table cells required
- Detects very *narrow column gaps*
- Handles *sparse tables*
- Recognizes *row- and/or column spanning cells*
- Uses only *common heuristics* and *does not require domain specific rules*
- Hence *no training or learning* phase and *no test set* required

- The evaluation is primarily *based on the geometry of word segments*. Hence it is *language independent*, deals with *low quality OCR* and *tolerates recognition errors*
- *T-Recs can be tested through a WWW interface* which is accessible with most web browsers: `http://www.dfki.uni-kl.de/∼kieni/t_recs/`

# 2 TABLE BOUNDARY EVALUATION

## 2.1 Subjective Performance Exploration

When testing the system on regular text pages which occasionally contain some tables, we obtain a very high recognition recall and also a high precision [1]. But there is another very important domain of documents containing tables, which is the domain of business letters in a purchasing environment (*offers*, *orders*, *order confirmations*, *delivery notes* and most important: *invoices*). The recognition and analysis of the tables in those documents is of high interest for office automation systems since they carry the most important information which moreover detracts itself from conventional text analysis techniques.

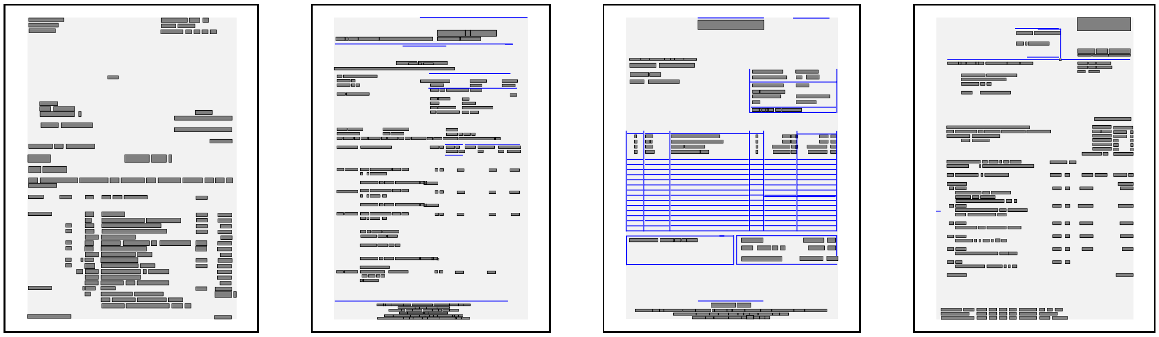

Figure 3: Sample letters with non-tabular layout objects in horizontal neighborship

When applying *T-Recs* to such letters we encounter a specific kind of error: it is the exact determination of the beginning and the end of the table boundaries. The simple heuristics which were implemented in the early prototype assumed to have some table-like structures wherever two or more text blocks occur in a horizontal neighborship. But the layout of business letters contains various logical object which are not tables but still stand in a left-of/right-of relation (see Figure 3). Examples herefore are the *recipient* and the *date field*; the *our sign*, *your sign* field or even the preprinted *company specific informations*. These objects not only mislead *T-Recs* to assume a table where there is no table. It also distorts the proper determination of the table boundaries by merging non-tabular elements with tabular elements.

We already addressed these problems of *T-Recs* when applied to business letters before [3]. There we proposed the application of the *Anastasil* system as a filter for *T-Recs*Ḃut this combination reduced the universal applicability of our overall system since *Anastasil* needs to be trained on sample letters of its final domain.

## 2.2 Extended Heuristics for Table Localization

The intention of the approach proposed here is to keep up the domain independance which is one of the strengths of *T-Recs*. To do so, we collected a series of typical features of tables. Most important here is the fact that the cells of one column are typically *aligned* to either left or right edge or that they are vertically centered. The cells of a column moreover tend

[1] Currently, we do not use objective benchmarking methods for the evaluation of the recognition accuracy. But we like to mention that this is one of our ongoing research topics.

to have *similar sizes* and *similar shape* (ratio of width to height). A small distance between blocks gives even further evidence.

Based on these features, we built a parametrized evidence function *incolumn*$(a, b)$ which accumulates a kind of index value for vertically adjacent blocks $a$ and $b$. This index tells how likely these cells belong to one table column.

The cluster of blocks that build a candidate table is now built in two steps: First determine the columns as vertical ranges of cells (based on the *incolumn*-function and some threshold values). Figure 4 visualizes this relation for a sample letterhead (left and center) and for the cells of a more intuitive table (right) with some unrestrictive thresholds. For these column candidates we can further determine an overall evidence *col_indx*.

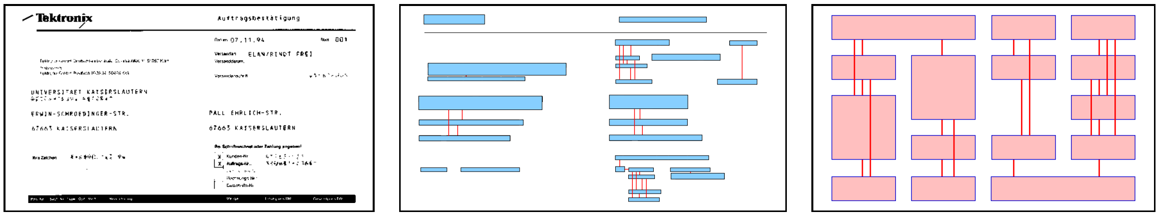

Figure 4: Visualization of the *incolumn*-predicate for a letter head and a regular table

In a second step, we determine the columns of the overall table based on the column candidates. Again we make use of typical features which are *close distance* and *horizontal alignment*. Together with the column index *col_indx* and given threshold values we determine the set of columns that belong to one table. Blocks that are not linked to any column but still reside inside the calculated bounding box will also be bound to the table structure.

## 3 CONCLUSION AND OUTLOOK

As the column aggregations of step 1 might have different vertical expansions, it is questionable where to start and where to end the overall table structure. We propose to generate all alternate tables based on the different expansions of the individual columns. For each alternative we evaluate an overall evidence score based on the vertical and horizontal alignment measures. While positive features have positive influence to the score, misalignments will have a negative effect. The best hypothesis will be selected based on the highest score.

The above mentioned postprocesing steps are currently about to be implemented in our prototype system. The effect upon the recognition precision can only be estimated. But besides improving the recognition process itself, we currently also develop a benchmarking environment, consisting of ground truthing frontend, a document corpus (including ground truth) and appropriate comparison mechanisms. This environment will grant an objective evaluation of new processing steps towards the overall result.

## References

[1] Thomas Kieninger. Table structure recognition based on robust block segmentation. In *Proc. of the fifth SPIE Conference on Document Recognition, San Jose, California*, January 1998.

[2] Thomas Kieninger and Andreas Dengel. A paper-to-html table converting system. In *Proc. of Document Analysis Systems - DAS 98, Nagano, Japan*, November 1998.

[3] Thomas Kieninger and Andreas Dengel. Table recognition and labeling using intrinsic layout features. In *Proc. of the first International Conference on Advances in Pattern Recognition - ICAPR 98, Plymouth, UK*, November 1998.