# A Trainable, Single-Pass Algorithm for Column Segmentation

Son Sylwester
*Concordia College, Seward, NE*, syl@seward.ccsn.edu

Sharad C. Seth
*University of Nebraska-Lincoln*, seth@cse.unl.edu

Proceedings of the Third International Conference on Document Analysis and Recognition, 1995, Volume 2.
doi: 10.1109/ICDAR.1995.601971

# A Trainable, Single-Pass Algorithm for Column Segmentation

Don Sylwester
Computer Science Department
Concordia College, Seward, NE
syl@seward.ccsn.edu

Sharad Seth
Department of Comp Sci & Eng
University of Nebraska - Lincoln, NE
seth@cse.unl.edu

## Abstract

*Column segmentation logically precedes OCR in the document analysis process. The trainable algorithm described here, XYCUT, relies on horizontal and vertical binary profiles to produce an XY- tree representing the column structure of a page of a technical document in a single pass through the bit image. Training against ground truth adjusts a single, resolution independent, parameter using only local information and guided by an* edit distance *function. The algorithm correctly segments the page image for a (fairly) wide range of parameter values, although small, local and repairable errors may be made, an effect measured by a* repair cost *function.*

**Keywords**: Column segmentation, decolumnization, XY tree, profiles

## 1 Introduction

Column segmentation is an essential part of any document analysis process. The algorithm described here, XYCUT, attempts column segmentation of page images guided by a single parameter based on a nearly universal feature of page layout design, that column gaps are wider than word and character gaps. Further, this single parameter is expressed in relative terms and is therefore independent of the scanning resolution. The value of the parameter is established through training against ground truth. Our experience with the cost function we have developed suggests that there is a relatively wide range for the parameter that yields a correct column segmentation, making the process reasonably robust.

Our algorithm has several novel features. The page is decolumnized in a single pass, from the top of the page to the bottom, with no backtracking.

Pavilidis and Zhou [1], on the other hand, employ a bottom-up approach that also focuses on locating column gaps but uses absolute parameters and horizontal smearing. Their approach is less skew sensitive than ours. Baird [2] also focuses on the white space in the layout but employs computational geometry to enumerate maximal white rectangles (covers) in the page image which are then unified in a fixed order. The unification process is stopped using an empirically derived rule.

Note that while our approach is based on finding cuts in binary profiles it is not necessary that these cuts reflect syntactic cuts in the page layout. Finally, our algorithm examines each pixel location in the page image approximately 5 times, suggesting the efficiency of this approach.

An up-to-date review of a variety of column segmentation methods may be found in the recent tutorial text [3], and a more complete summary of our present work is contained in [4].

## 2 Column Segmentation

Column segmentation begins with the identification of column gaps in horizontal slices of the page image. Once a horizontal slice has been segmented the components are then assembled into the growing XY-tree that stores the column structure.

**Identification of Column Gaps**. The specific feature we are looking for is a gap that is relatively wide compared to the height of the text lines on either side. The measure of the gap size is taken to be the ratio of the width of the gap to the minimum height of the text lines in the block on either side, a height measured by the binary profile of the blocks. This definition produces a relative measure that is independent of the resolution of the scanned image. This measure is then compared to a threshold and all gaps whose relative gap size exceeds the threshold are treated as column gaps.

The threshold value is identified through training against a test set for which ground truth in the form

0-8186-7128-9/95 $4.00 © 1995 IEEE

of an XY-tree with isolated text lines as the leaves is available. The evaluation process, performed by inspection for this experiment, will be automated when machine-readable ground truth has been developed.

**Construction of the XY-tree.** The column structure of the document is stored in an XY-tree. Each node in the tree represents a block of pixels in the page image with nodes in alternate levels being formed most recently by horizontal cuts (forming *slices*) or vertical cuts (forming *columns*). The leaves of the tree are the isolated text lines that make up the columns and are all slice blocks as is the root. The current implementation of XYCUT does not distinguish text and non-text blocks so non-text blocks are also found at the leaf level.

The horizontal binary profile of the document page is used to cut the page into one or more horizontal slices. Note that the algorithm does not depend on two adjacent columns having their text lines aligned. The vertical binary profile of each slice then locates potential column gaps for comparison with the threshhold. The individual blocks of the now segmented slice are recursively sliced to form a forest of columnar segments of the slice.

The horizontal overlap between the segments in the current slice and the segments of the currently visible column structure, maintained as the tree is being grown, are compared. The segments from the new slice are merged into the growing tree following a careful and somewhat involved case analysis.

## 3 Training

A column containing text has been correctly segmented if two conditions are met: a) Every leaf node contains only a single text line or a portion of a single text line, b) Every parent of a leaf node contains only complete text lines or portions of single text lines. The types of errors that violate these conditions fall into two broad classes, *oversegmentation* and *undersegmentation*. Within these classes we may further identify errors which affect the correctness of the segmentation and those that affect its quality but still describe a correct reading order for the column.

**Correct Segmentation: Edit Distance.** The use of a simple comparison of the measure of a column gap to a single threshold invites two kinds of errors which result in incorrect column segmentation.

*Undersegmentation:* In the first case an actual column gap may not be identified as a column gap and the text from two text lines in different columns may be merged in a single node of the XY-tree. Errors of this type begin to occur as the value of the threshold is increased.

*Oversegmentation:* In the second case gaps between words in vertically adjacent text lines are incorrectly identified as column gaps. Errors of this type begin to occur as the value of the threshold is decreased. If erroneous column gaps in vertically adjacent text lines overlap they will cause the XY-tree merge procedure to incorrectly construct columns from portions of two or more text lines.

We have found in looking at document pages from several journals that instances of these two types of errors occur for non-overlapping ranges of threshold values, and that for a substantial interval between these non-overlapping ranges of threshold values neither error occurs. This interval represents 48% of the value of the parameter for test pages chosen from IBM J Res Dev and 15% of the parameter value for test pages chosen from PAMI.

To measure the effect of each of these errors we apply an *edit cost* function, inspired by the string edit model used by ISRI [5] to evaluate the accuracy of OCR methods. We compare the reading order for each column with an inorder traversal of that portion of the XY-tree which should correspond with that column and determine the insert and delete operations necessary to correct the reading order from the traversal. These operations are weighted by the areas of the nodes involved and the total area is the edit cost, normalized by the (loose) upper bound of twice the bounding box of the page image.

**Quality of Segmentation: Repair Cost.** There are two types of errors which do not affect the reading order for a correctly identified column.

*Undersegmentation:* The first error type occurs when two vertically adjacent text lines do not have a horizontal cut between them visible in the horizontal binary profile. This occurs most often because of an overlap between a descender from the upper line and an ascender from the lower line but it can also result from a relatively high skew angle. If the leaf node contains only two or more complete text lines from the column then the column may still be read correctly.

*Oversegmentation:* In the second type gaps, presumably gaps between words, are mistakenly identified as column gaps. This often occurs, for instance, when justification forces large gaps between words in order to maintain alignment with the right edge of the column. If these extra gaps merely segment a single text line into multiple components then the column may still be read correctly.

Since neither difference contributes to an error in the reading order for the column it is useful to define a *repair cost*, a cost that provides a distance measure between the correct but flawed segmentation tree and the ground truth. A repair cost of zero represents identical segmentation and ground truth trees.

In the case of extra column gaps we define a *glue* operation with the cost equated to the number of extra gaps between leaf nodes which need to be glued together. In the case of a missing horizontal cut we define a *snip* operation with the cost equated to the number of additional horizontal cuts that must be made to match ground truth.

The number of glue operations is bounded loosely by the number of words on the page and the number of snip operations is bounded by the number of text lines.

Note that oversegmentation is sensitive to the value of the relative-gap-size threshold, but that undersegmentation, which depends only on the use of horizontal profiles to locate cuts, is not.

## 4 Results

**Training**. The algorithm was trained separately on two technical journals, *IBM Journal of Research and Development* and *IEEE Pattern Analysis and Machine Intelligence*, with three pages selected from each journal. The results are summarized in Table 1. which includes both the parameter ranges for each page over which the edit-distance is zero as well as the repair cost at the centers of the intervals.

Table 1: Training Summary

| Journal Page | Interval for Zero | Edit Distance | | Repair Cost on Interval at Midpt |
|---|---|---|---|---|
| | Left | Right | Range | |
| IBM J. R&D | | | | |
| IBM_N1 | 1.29 | 2.84 | 1.55 | 0 |
| IBM_N2 | 1.39 | 2.27 | 0.88 | 0 |
| IBM_T1 | 1.19 | 2.71 | 1.52 | 0 |
| Intersection | 1.39 | 2.27 | 0.88 | |
| Intersection midpoint: 1.83 | | | | |
| IEEE PAMI | | | | |
| PAMI_N1 | 1.00 | 1.45 | 0.45 | .012 |
| PAMI_N2 | 0.98 | 1.34 | 0.36 | .026 |
| PAMI_T1 | 1.15 | 1.54 | 0.39 | .006 |
| Intersection | 1.15 | 1.34 | 0.19 | |
| Intersection midpoint: 1.24 | | | | |

Figure 1 is a graph, for a single page image, of the normalized edit distance and the normalized repair cost as a function of the gap-size parameter. Note that the repair cost is defined only over the interval where the edit cost is zero.

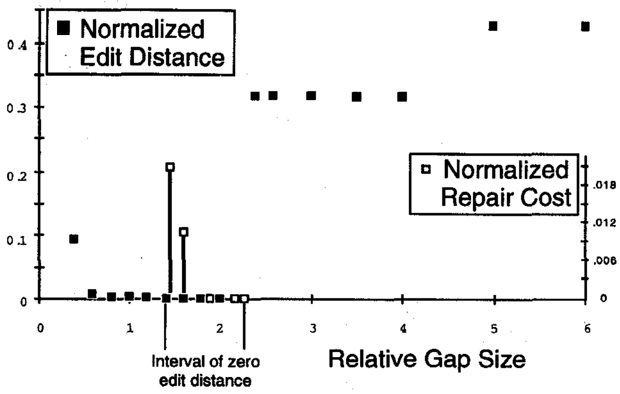


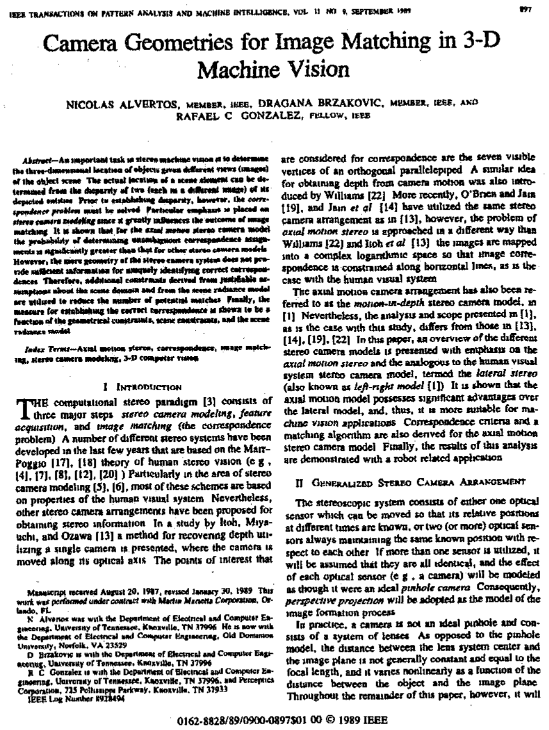
Figure 1: Normalized edit distance and normalized repair cost for page IBM_N2

Table 2: Testing Summary

| | | IBMJ | PAMI |
|---|---|---|---|
| Test Pages | | 7 | 7 |
| Correct Pages | | 7 | 5 |
| Repair Cost | Min | 0 | 16 |
| | Max | 1 | 259 |
| Error pages | | 0 | 2 |
| Edit Cost | Avg cost | - | .012 |
| | Avg operations | - | 1.5 |

**Testing**. The algorithm was applied to a test set consisting of 14 page images chosen from the two journals. For each image the edit distance and the repair cost were computed for the gap-size parameter selected for each journal during the training process. Table 2 displays the results. Figure 2 shows a one of the test pages and Figure 3 shows the main features of the XY-tree (levels 3, 4, and 6) including examples of several repairable errors. The skew angle for this page was about 0.5 degree.

The algorithm successfully decolumnized all 7 pages from IBM J Res Dev with only one instance of a required glue operation. The algorithm successfully decolumnized 5 of the 7 pages from PAMI. While the actual columns of the 2 unsuccessful pages were properly identified, three instances of columns formed from portions of text lines occurred.



IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL 11 NO 9, SEPTEMBER 1989 897

Camera Geometries for Image Matching in 3-D Machine Vision

NICOLAS ALVERTOS, MEMBER, IEEE, DRAGANA BRZAKOVIC, MEMBER, IEEE, AND
RAFAEL C GONZALEZ, FELLOW, IEEE

*Abstract*—An important task in stereo machine vision is to determine the three-dimensional location of objects given different views (images) of the object scene The actual location of a scene element can be determined from the disparity of two (each in a different image) of its depicted entities Prior to establishing disparity, however, the *correspondence problem* must be solved Particular emphasis is placed on *stereo camera modeling* since it greatly influences the outcome of image matching It is shown that for the *axial motion* stereo camera model the probability of determining unambiguous correspondence assignments is significantly greater than that for other stereo camera models However, the mere geometry of the stereo camera system does not provide sufficient information for uniquely identifying correct correspondences Therefore, additional constraints derived from justifiable assumptions about the scene domain and from the scene radiance model are utilized to reduce the number of potential matches Finally, the measure for establishing the correct correspondence is shown to be a function of the geometrical constraints, scene constraints, and the scene radiance model

*Index Terms*—Axial motion stereo, correspondence, image matching, stereo camera modeling, 3-D computer vision

I INTRODUCTION

THE computational stereo paradigm [3] consists of three major steps *stereo camera modeling*, *feature acquisition*, and *image matching* (the correspondence problem) A number of different stereo systems have been developed in the last few years that are based on the Marr–Poggio [17], [18] theory of human stereo vision (e g , [4], [7], [8], [12], [20] ) Particularly in the area of stereo camera modeling [5], [6], most of these schemes are based on properties of the human visual system Nevertheless, other stereo camera arrangements have been proposed for obtaining stereo information In a study by Itoh, Miyauchi, and Ozawa [13] a method for recovering depth utilizing a single camera is presented, where the camera is moved along its optical axis The points of interest that are considered for correspondence are the seven visible vertices of an orthogonal parallelepiped A similar idea for obtaining depth from camera motion was also introduced by Williams [22] More recently, O'Brien and Jain [19], and Jain *et al* [14] have utilized the same stereo camera arrangement as in [13], however, the problem of *axial motion stereo* is approached in a different way than Williams [22] and Itoh *et al* [13] the images are mapped into a complex logarithmic space so that image correspondence is constrained along horizontal lines, as is the case with the human visual system

The axial motion camera arrangement has also been referred to as the *motion-in-depth* stereo camera model, in [1] Nevertheless, the analysis and scope presented in [1], as is the case with this study, differs from those in [13], [14], [19], [22] In this paper, an overview of the different stereo camera models is presented with emphasis on the *axial motion stereo* and the analogous to the human visual system stereo camera model, termed the *lateral stereo* (also known as *left-right model* [1]) It is shown that the axial motion model possesses significant advantages over the lateral model, and, thus, it is more suitable for *machine vision* applications Correspondence criteria and a matching algorithm are also derived for the axial motion stereo camera model Finally, the results of this analysis are demonstrated with a robot related application

II GENERALIZED STEREO CAMERA ARRANGEMENT

The stereoscopic system consists of either one optical sensor which can be moved so that its relative positions at different times are known, or two (or more) optical sensors always maintaining the same known position with respect to each other If more than one sensor is utilized, it will be assumed that they are all identical, and the effect of each optical sensor (e g , a camera) will be modeled as though it were an ideal *pinhole camera* Consequently, *perspective projection* will be adopted as the model of the image formation process

In practice, a camera is not an ideal pinhole and consists of a system of lenses As opposed to the pinhole model, the distance between the lens system center and the image plane is not generally constant and equal to the focal length, and it varies nonlinearly as a function of the distance between the object and the image plane Throughout the remainder of this paper, however, it will

Manuscript received August 20, 1987, revised January 30, 1989 This work was performed under contract with Martin Marietta Corporation, Orlando, FL

N Alvertos was with the Department of Electrical and Computer Engineering, University of Tennessee, Knoxville, TN 37996 He is now with the Department of Electrical and Computer Engineering, Old Dominion University, Norfolk, VA 23529

D Brzakovic is with the Department of Electrical and Computer Engineering, University of Tennessee, Knoxville, TN 37996

R C Gonzalez is with the Department of Electrical and Computer Engineering, University of Tennessee, Knoxville, TN 37996, and Perceptics Corporation, 725 Pellissippi Parkway, Knoxville, TN 37933

IEEE Log Number 8928494

0162-8828/89/0900-0897$01 00 © 1989 IEEE

Figure 2: Page PAMI_T1

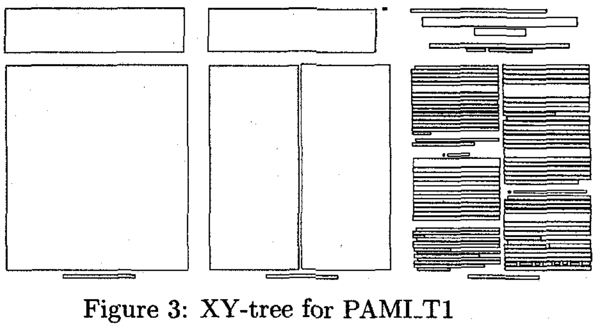


Figure 3: XY-tree for PAMI_T1

## 5 Conclusions

There is a significant range of values for the threshold parameter for which the resulting XY-tree generates correct reading order for the text within its columns.

The algorithm shares the common limitations of all methods based on binary profiles. The images must be relatively noise-free and have small skew angles, but there are a lot of page images within this domain (see, for example [6]), especially if a preprocessing step adjusts skew and removes identifiable noise.

The string edit cost model used to evaluate OCR methods can be adapted to provide an edit distance function.

**Acknowledgements**. The authors acknowledge George Nagy for many helpful conversations and Mahesh Viswanathan for supplying the cleaned, deskewed page images, several by way of the IEEE database.

## References

[1] T. Pavlidis and J. Zhou. Page segmentation and classification. *CVGIP: Graphical Models and Image Processing*, 54(6):484–496, 1992.

[2] H. Baird. Background structure in document images. In *Proceedings of IAPR Workshop on Document Analysis*, pages 1–17, 1992.

[3] L. O'Gorman and R. Kasturi. *Document Image Analysis*. IEEE Computer Society Press, 1995.

[4] D. Sylwester and S. Seth. A trainable, single-pass algorithm for column segmentation. Univ. Nebraska-Lincoln, Dep. of Comp Sci and Eng, Tech. Rep. UNL-CSE-95-003, April 1995.

[5] J. Kanai, S. Rice, and T Nartker. A preliminary analysis of automatic zoning. In *ISRI Annual Report*, pages 35–45. 1994.

[6] S. Chen, M. Y. Jaisimha, J. Ha, and R.M. Haralick. Reference manual for UW English Document Image Database I, 1993.