Active Document Layout Synthesis

Xiaofan Lin
Imaging Systems Laboratory
HP Laboratories Palo Alto
HPL-2005-106
May 25, 2005*

E-mail: xiaofan.lin@hp.com

variable data printing, constraint solving, document layout synthesis, table formatting, Simplex

Document layout analysis has been researched for many years. However, there is little work on the reverse of document layout analysis: document layout synthesis, whose goal is to generate logically correct and aesthetically appealing layout given the text/image contents and the flexible layout template. This paper introduces a new automatic document layout synthesis method, which can actively pursue the optimal text block height-width tradeoff simultaneously with the block position adjustment. The key idea is to use a cluster of linear functions to approximate the nonlinear height-width curve. Then two-pass Simplex algorithm is used to solve the layout synthesis problem. This method has a wide range of applications, such as Variable Data Printing (VDP), automatic table formatting, and automating document layout ground truth generation.

* Internal Accession Date Only
To be published in and presented at the 8th International Conference on Document Analysis and Recognition, 29 August - 1 September 2005, Seoul, Korea

Approved for External Publication

Active Document Layout Synthesis

Xiaofan Lin
Hewlett-Packard Laboratories
1501 Page Mill Road, MS 1203, Palo Alto, CA 94304, USA
Email: xiaofan.lin@hp.com

Abstract

Document layout analysis has been researched for many years. However, there is little work on the reverse of document layout analysis: document layout synthesis, whose goal is to generate logically correct and aesthetically appealing layout given the text/image contents and the flexible layout template. This paper introduces a new automatic document layout synthesis method, which can actively pursue the optimal text block height-width tradeoff simultaneously with the block position adjustment. The key idea is to use a cluster of linear functions to approximate the nonlinear height-width curve. Then two-pass Simplex algorithm is used to solve the layout synthesis problem. This method has a wide range of applications, such as Variable Data Printing (VDP), automatic table formatting, and automating document layout ground truth generation.

1. Introduction

Much research has been conducted on document layout analysis [1][2], one important area in document analysis and recognition. Document layout analysis (DLA) starts from the instantiated layout (usually a bitmapped document image captured by a scanner or a digital camera) and attempts to locate and classify the individual blocks. The text blocks can then be converted back to words using Optical Character Recognition (OCR). The reverse of document layout analysis also poses an interesting research problem: how to decide the most appropriate dimension and position of each block given the text and image contents. In this paper we use the term "document layout synthesis" (DLS) to describe this task. Figure 1 compares DLA with DLS. While document layout analysis is a pattern recognition process, document layout synthesis is an optimization process and constraint solving techniques [3] provide the powerful tools. DLS plays an important role in highly customized Variable Data Printing (VDP) applications. For example, the contents of individual blocks are dynamically retrieved from a Customer Relationship Management (CRM) system and then automatically composed into brochures or catalogs using DLS technology.

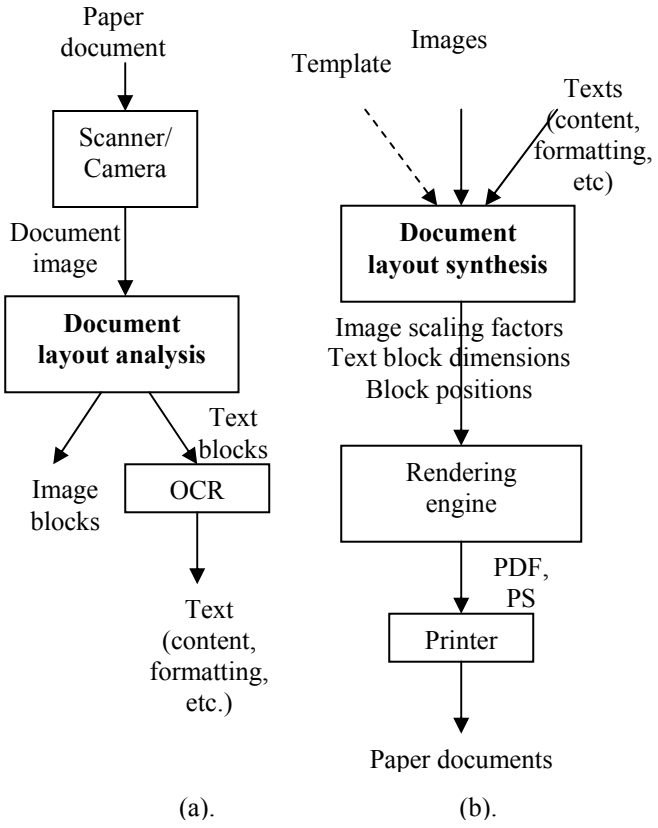

Figure 1: Comparison of document layout analysis (a) and synthesis (b)

The goal of DLS is to create logically correct (for example, two text blocks should not overlap) and aesthetically appealing (for example, balanced and aligned) layouts. There is some previous work on DLS. Jacobs et al. [4] introduced an adaptive document layout system that can automatically select the best template for given contents. Johari et al. [6] created a special-purpose pagination and layout system for yellow pages. Badros et al. [5] proposed a constraint extension to Scalable Vector Graphics (SVG) to enable interactive

graphics on the Web. In the existing work, however, the width of each text block is either fixed or completely determined by the template (for example, the column width of a three-column page can be calculated given the page width). If the block width is determined, the block height can be determined and then the text block can be treated in the same way as an images block in layout synthesis (see Figure 2.a). In this paper we present an active DLS method. Instead of fixing the text block width through a rigid template, the template defines the relative positions of individual blocks without fixing the text block width. A multi-linear height-width model is built for each text block and then is incorporated into a two-pass layout synthesis process (see Figure 2.b). Section 2 presents the multi-linear text modeling, which is the basis of the active DLS. Section 3 is devoted to the two-pass Simplex algorithm. Section 4 discusses the experimental results using a concrete example. In Section 5, active DLS is applied to table formatting problem. Section 6 gives a summary.

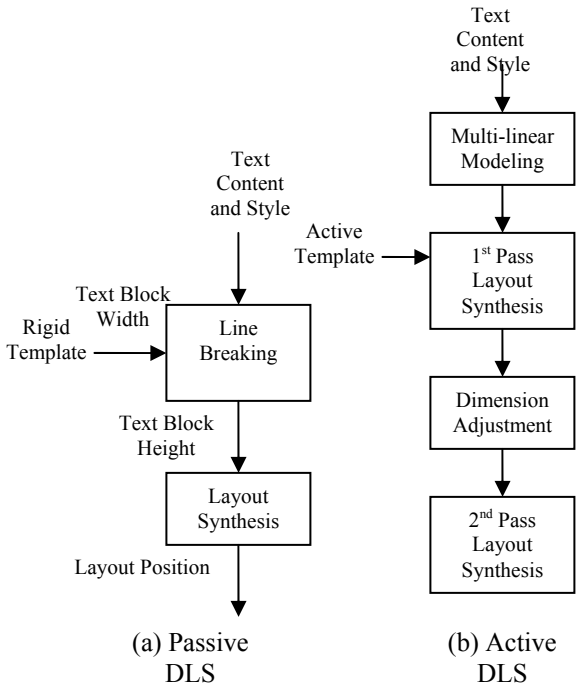

Figure 2: Passive DLS vs. Active DLS

2. Multi-linear Text Modeling

Assume there are n rectangular content blocks on a page: B1, B2, …, Bn, each of which will hold text, image, or graphics. Each block has its intrinsic geometric attributes, such as the width and height, expressed as a vector S(Bi). On the page layout, each block also occupies a position, which can be specified as the coordinates P(Bi) of the top left corner. If S(Bi) is fixed in the layout synthesis (as the case in passive DLS), we only need to find out a set of P(Bi) to optimize a layout quality function. When each block is rectangular, this problem can be formulated as a set of linear constraints and solved with Simplex algorithm.

In active DLS, we want to also adapt S(Bi) (for example, changing the width of a text block) together with P(Bi), things will be much more complicated. The biggest challenge in active DLS is the non-linear step-wise nature of the text block height-width relationship, as shown in Figure 3. When the width changes continuously, the height will only "jump" from one value to another when the number of lines changes. This characteristic makes it impossible or very inefficient to apply standard constraint solving techniques such as Simplex [7]. So the first step in active DLS is to model the text height-width relationship as multiple linear constraints to enable Simplex.

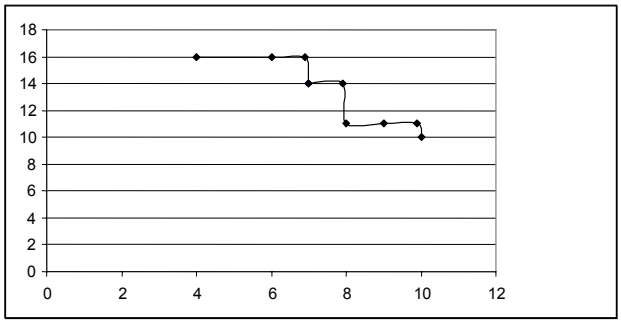

Figure 3: Non-linear relationship between the width (x-axis) and the height (y-axis) of a rectangular text block, given the text content and font style

Heuristically, the height-width relationship roughly follows a hyperbolic function: height*width=constant. So the first step is to formulate such hyperbolic function using polynomial regression. Through accessing the text rendering engine (for example, Apache FOP [10]), we can get several actual pairs of (width, height), as shown in Table 1. With the sampling data, the optimal coefficients k and b in the following hyperbolic model can be calculated using regression techniques to minimize the mean square error (MSE):

$$h=k/w+b \quad (1)$$

Based on the data of Table 1, k and b are calculated to be 8360.6 and -1.04 respectively. Equation 1 is a non-linear function and still cannot be handled by Simplex. So a number of sampling points are located on h=k/w+b across the maximal allowed range of w. For example, we can find 20 sampling points with w in the range of [50 points, 500 points]. It is also preferred that the intervals between the heights of the sampling points are

constant. Then the height-width relationship of each text block can be expressed a cluster of linear constraints (see Figure 4):

F(width,i)= h[i]+(h[i]-h[i-1])*(width-w[i-1])/(w[i]-w[i-1]) **(2)**
height>=F(width,i), i=1,…, 19

Table 1: Lookup table for height-width relationship

Width (points)	200	…	350	…	500
Height (lines)	41		23		16

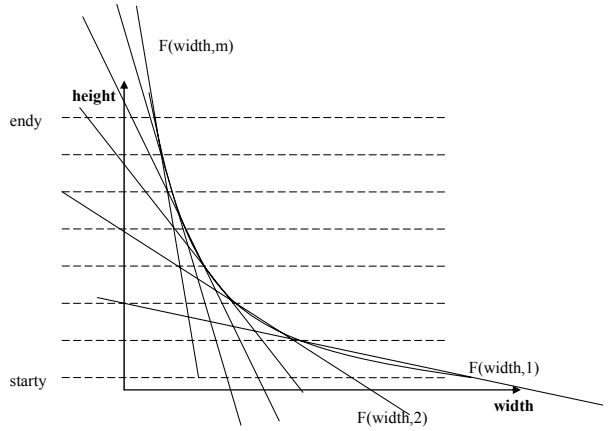

Figure 4: Create a cluster of linear functions to cover the hyperbolic function

3. Two-pass Constraint Solving

With the multi-linear height-width modeling, we can use Simplex to solve the layout generation problem. The layout is solved in two passes. The first pass decides the optimal width of each text block. Then through actual line-breaking, we can get the height of each text block. The second pass then decides the final positions of the text block as well as the positions and sizes of the image blocks.

3.1. First-pass Simplex

The Cassowary constraint solver [3], which is an improved version of Simplex, is adopted and CSVG [5] is used to express linear equality and inequality constraints. The constraints to the first-pass Simplex consist of two parts. The first part comes from the active layout template and relates to the relative positions of different blocks. Here are a couple of examples:

<constraint rule="block3_left >= block2_right+10" strength= "required"/>

<constraint rule="block2_left >= block1_right+10" strength= "required"/>

The above two constraints dictate the relative horizontal positions of Block 1, 2, and 3: Block 3 is to the right of Block 2, and Block 2 is to the right of Block 1. The second part is produced by the multi-linear text modeling process and formulates the text block height-width relationship:

<constraint rule="block1_height>=67.5+(block1_width - 1059.5) *(-0.067)" strength="required"/>

<constraint rule="block1_height>=101.9+(block1_width-543.6) *(-0.19)" strength="required"/>

…

Then the constraint solver can find the optimal values for the variables, such as:

block1_width = 172.8 block1_height = 254.9

block1_left = 50.0 block1_top = 171.7

block2_width = 134.4 block2_height = 322.5

block2_left = 232.8 block2_top = 171.7

Due to the approximate nature of the multi-linear modeling, the text block heights are just estimates. So the actual line-breaking is carried out on each text block based on the calculated width. Then we can get the accurate text block heights. In the above example, the heights are adjusted to:

block1_height = 202.5 block2_height = 310.5

3.2. Second-pass Simplex

In the second-pass Simplex, instead of multi-linear constraints for each text block, we fix the size of each text block. In the above example, the sizes for Block 1 and 2 are governed by:

<constraint rule="block1_width=172.8" strength ="required"/>

<constraint rule="block1_height=202.5" strength="required"/>

<constraint rule="block2_width=134.4" strength="required"/>

<constraint rule="block2_height=310.5" strength="required"/>

The other constraints with regard to the relative positions maintain unchanged. The second-pass Simplex can then finalize the layout.

4. Experimental Results

This section gives an example to show the effectiveness of the proposed method. The active template contains the following requirements:

1) B2 is to the right of B1.
2) Image block B1 has the same height as the text block B2.
3) The height and width of B3 are the same, or B3 should be a square.
4) B3, B4, and B5 are below B1 and B2.

5) The tops of B3, B4 and B5 are vertically aligned.
6) B6 is horizontally aligned with B3
7) The images are allowed to scale proportionally with the aspect ratio unchanged.
8) Minimize the total height of the occupied space to achieve a compact page.

Figure 5 shows the intermediate layout after the first-pass Simplex. The layout is almost perfect, with the exception of an unnecessary gap in the middle. This imperfection is due to the estimate errors introduced by the multi-linear modeling. Figure 6 is the final layout after the second-pass Simplex. As expected, the unnecessary gap is removed.

It is worth mentioning that Requirements 2 and 8 are quite subtle. The layout generation algorithm intelligently chooses the ratio between the widths of B4 and B5 to make sure that they have roughly the same height. Mathematical derivation shows that it is the way to minimize the height of the occupied space. It also selects the optimal scaling factor for B1 image to make sure that B2 has the same height as B1. Mathematical analysis also shows that it is a quadratic equation which can have 0, 1, and 2 solutions depending on the aspect ratio of B1 and the area of B2. When it has two solutions, only the first one will result in the minimal space height. With the proposed method, we can find the optimal layout systematically without explicitly deriving and solving the underlying equations.

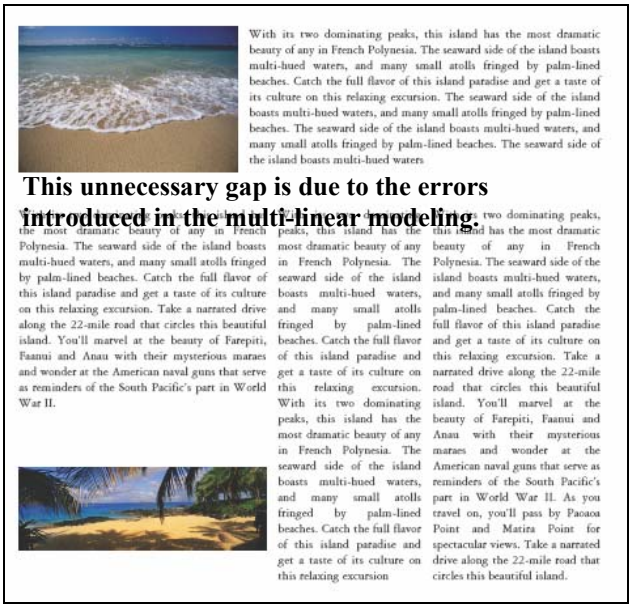

Figure 5: Intermediate layout after the first-pass Simplex

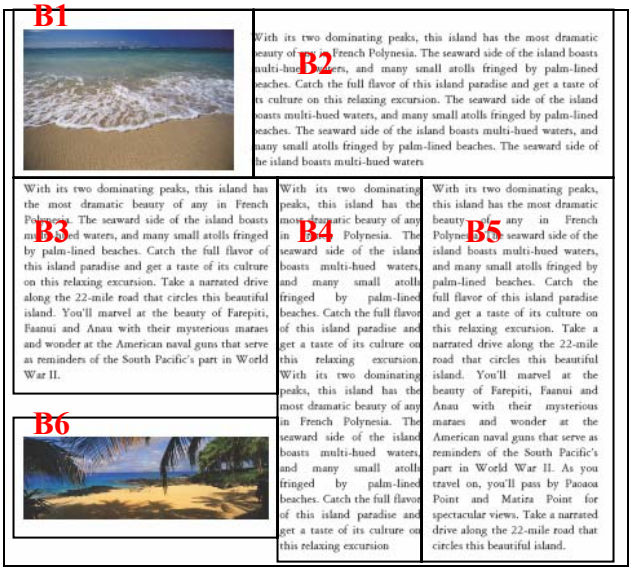

Figure 6: Final layout after the second-pass Simplex (The gap between the top and bottom is eliminated.)

5. Application to Table Formatting

A common task in publishing is table formatting, whose goal is to decide the optimal dimensions of table cells given the contents and the table's grid structure. The optimal table layout is usually the one leading to the minimal table height and satisfies other aesthetical criteria such as balance. Formatting a complex table can be a challenge even for humans because the optimal table layout can only be achieved by making the right choice for every cell. Existing methods on table formatting, such as those proposed by Wang in his thesis [8], are various heuristic search methods with worse-case exponential and average/best-case polynomial computational time with reference to the number of cells, columns, and rows. Fortunately the active DLS method can be easily applied to table formatting:

Step 1: Describe the table in an XML-based language. As shown in Figure 7, x1, x2, … , xm specify the ordered x-coordinates that a cell can start or end, and y1, y2, …, yn specify the ordered y-coordinates that a cell can start or end. Then the table can be described in the following XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<table rows="5" columns="6" starty="50" startx="100"
endx="500" gapx="5" gapy="5" pagewidth="600"
pageheight="800">…
<cell startx="1" starty="1" endx="1" endy="2"> Time
</cell>
<cell startx="1" starty="3" endx="1" endy="3">
Morning 9:00-12:00
```

</cell>…

<table> element specifies global parameters for the whole table, such as the numbers of columns and rows, page dimensions, etc. <cell> element specifies each cell: startx, starty, endx, and endy delimit the boundary of the cell. Other table description languages may be used.

Step 2: Use a converter program to transform the XML table description into active layout template required by active DLS. This template will define the relative positions of individual cells. Here are some constraints generated by the converter:

```
<constraint rule="y1>=50.0" strength="required"/>
<constraint rule="y2>=y1" strength="required"/>
<constraint rule="y3>=y2" strength="required"/>
<constraint rule="y4>=y3" strength="required"/>
<constraint rule="y5>=y4" strength="required"/>
<constraint rule="y6>=y5" strength="required"/>
<constraint rule="y6=0" strength="strong"/>
<constraint rule="800.0>=y6" strength="required"/>
<constraint rule="x1>=100.0" strength="required"/>
<constraint rule="x2>=x1" strength="required"/>
<constraint rule="x3>=x2" strength="required"/>
<constraint rule="x4>=x3" strength="required"/>
<constraint rule="x5>=x4" strength="required"/>
<constraint rule="x6>=x5" strength="required"/>
<constraint rule="x7>=x6" strength="required"/>
<constraint rule="500.0>=x7" strength="required"/>
<constraint rule="table11_left=x1" strength="required"/>
<constraint rule="x2>=table11_right+5.0"
strength="required"/>
<constraint rule="table11_top=y1" strength="required"/>
<constraint rule="y3>=table11_bottom+5.0"
strength="required"/>…
```

Step 3: Find the optimal layout using active DLS method. Figure 7 shows an example of the final table layout. It can be seen that active DLS wisely chooses the widths of different columns to achieve a compact table layout.

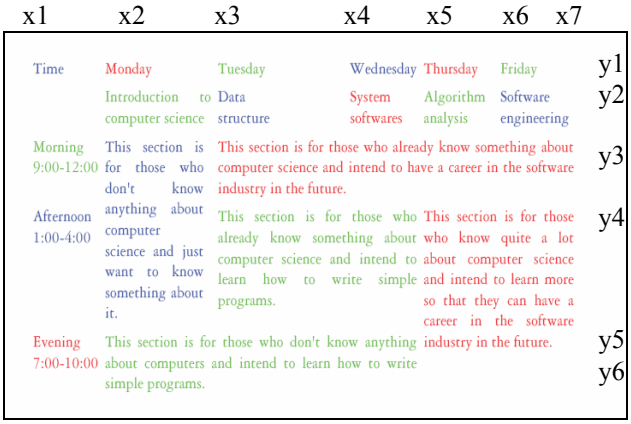

Figure 7: Table formatting results from active DLS

6. Conclusions

After comparing document layout synthesis with document layout analysis, this paper introduces a new automatic document layout synthesis method, which can actively pursue the optimal text block height-width tradeoff simultaneously with the block position adjustment. We have also shown that the proposed active DLS method can directly solve another special layout problem: table formatting. In addition to applications in VDP and table formatting, another use of DLS is to automate document layout ground truth generation, which is very valuable for quantitative evaluation and systematic training of DLA algorithms [9]. With DLS, we can automatically generate large number of layouts instead of manually generating the layouts using a word processor. In this sense, DLS can help to improve its DLA counterpart. This is definitely an interesting direction for our future research.

7. References

[1] F.M. Wahl, K.Y. Wong, and R.G. Casey "Block Segmentation and Text Extraction in Mixed/image Documents," *Computer Vision Graphics and Image Processing*, vol. 2, 1982, pp. 375-390.

[2] J. Shi, J. Malik, "Normalized Cuts and Image Segmentation," *IEEE Transactions on Pattern Analysis Machine Intelligence*, vol. 22, no. 8, 2000, pp. 888-905.

[3] G. Badros, A. Borning, "The Cassowary linear arithmetic constraint solving algorithm: Interface and implementation," Technical Report UW-CSE-98-06-04, University of Washington, Seattle, Washington, June 1998.

[4] C. Jacobs, W. Li, et al., "Adaptive Grid-based Document Layout," *ACM Transaction on Graphics*, vol. 22, no. 3, 2003, pp. 838-847.

[5] G. Badros, J. Tirtowidjojo, et al., "A Constraint Extension to Scalable Vector Graphics," *Proceedings of Tenth International World Wide Web Conference*, Hong Kong, May 2001.

[6] R. Johari, J. Marks, et al., "Automatic Yellow-Pages Pagination and Lahyout," Mitsubishi Electric Research Laboratory Technical Report TR-96-29, 1996.

[7] G. B. Dantzig, W. Orchard-Hays, "The product Form for the Inverse in the Simplex Method," *Mathematical Tables and Other Aids to Computation*, 8, 1954, pp. 64-67.

[8] X. Wang, "Tabular Abstraction, Editing, and Formatting," Ph.D. Thesis, University of Waterloo, 1996.

[9] T. Kanungo T, S. Mao, "Stochastic Language Model for Style-Directed Physical Layout Analysis of Documents," *IEEE Transactions on Image Processing*, vol. 12, no. 5, 2003, pp. 583-596.

[10] http://xml.apache.org/fop