Ensure Non-Overlapping in Document Layout Analysis

Costin-Anton Boiangiu
Polytechnic University of Bucharest
158 PUBLICATIONS **493** CITATIONS
SEE PROFILE

Ion I. Bucur
Polytechnic University of Bucharest
61 PUBLICATIONS **157** CITATIONS
SEE PROFILE

PDF OFF-PRINTS

Author(s):

BOIANGIU, C[ostin] - A[nton]; RADUCANU, B[ogdan]; PETRESCU, S[erban] & BUCUR, I[on]

This Publication has to be referred as:

Boiangiu, C. - A.; Raducanu, B.; Petrescu, S. & Bucur, I. (2009). Ensure Non-Overlapping in Document Layout Analysis (2009). 0327-0329, *Annals of DAAAM for 2009 & Proceedings of the 20th International DAAAM Symposium*, ISBN 978-3-901509-70-4, ISSN 1726-9679, pp 164, Editor B[ranko] Katalinic, Published by DAAAM International, Vienna, Austria 2009

Annals of DAAAM for 2009 & Proceedings of the 20th International DAAAM Symposium, Volume 20, No. 1, ISSN 1726-9679
ISBN 978-3-901509-70-4, Editor B. Katalinic, Published by DAAAM International, Vienna, Austria, EU, 2009
Make Harmony Between Technology and Nature, and Your Mind will Fly Free as a Bird
Annals of DAAAM International

ENSURE NON-OVERLAPPING IN DOCUMENT LAYOUT ANALYSIS

BOIANGIU, C[ostin] - A[nton]; RADUCANU, B[ogdan]; PETRESCU, S[erban] & BUCUR, I[on]

Abstract: *Document layout analysis is a process that attempts to break down the structure of a scanned document and extract important layout elements such as paragraphs, headlines, images and so on. After this process, there is a need to mark the identified elements through an encapsulating shape that is visually clearly defined and easy to manipulate. This paper describes a method for achieving this. The results are very accurate and the method has a high degree of stability.*
Key words: *clipping, conversion, marker, layout*

1. INTRODUCTION

There are many steps involved in the process of layout analysis. Raw preprocessing is needed to eliminate noise and align the page. Small image entities like characters are identified through a process called segmentation. Heuristically algorithms are then used to detect paragraphs, columns, tables, images, etc. Finally, this information has to be presented to a user in a friendly way that allows a flexible manipulation.

One problem that arises here is determining which shape to use to encapsulate a paragraph, for instance, such that the user sees exactly where the paragraphs is and its bounding shape is as simple as possible.

There is no study regarding this exact problem. It is often treated as a minor front-end design issue. The present paper will describe a method to compute for each layout element a simple shape to use as a marker for it and to allow document clipping, if needed.

2. METHOD

The shape computed by this method is a hybrid of the following: bounding rectangle, rectangular shape, oriented bounding box (OBB), convex hull, min max curve and the beta-shape. The **bounding rectangle** is the simplest shape possible. It is the minimum area rectangle that includes the layout element and that is also aligned with the page axis. The **rectangular shape** is ideal for textual paragraphs. It is constructed by the union of the rectangles that bound each line of the paragraph.

The **oriented bounding box (OBB) (**Edelsbrunner, 1987; Pratt, 2002) is the minimum area rectangle that includes the layout element. This rectangle can be of any orientation, because of this, it is harder to determine and often not preferred. In many applications the bounding box is aligned with the axes of the coordinate system, and it is then known as an axis-aligned bounding box (AABB).

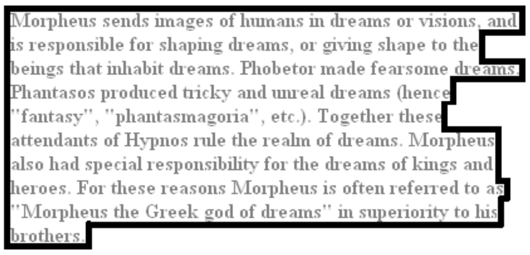
Morpheus sends images of humans in dreams or visions, and
is responsible for shaping dreams, or giving shape to the
beings that inhabit dreams. Phobetor made fearsome dreams.
Phantasos produced tricky and unreal dreams (hence
"fantasy", "phantasmagoria", etc.). Together these
attendants of Hypnos rule the realm of dreams. Morpheus
also had special responsibility for the dreams of kings and
heroes. For these reasons Morpheus is often referred to as
"Morpheus the Greek god of dreams" in superiority to his
brothers.

Fig. 1. Rectangular shape

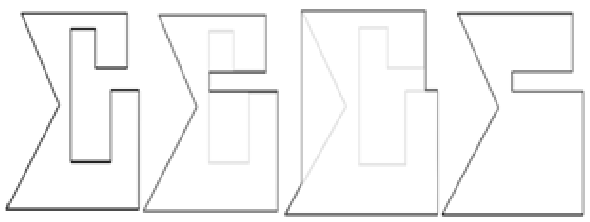

Fig. 2. Min-max shape: a) point set, b) min-max-x, c) min-max-y, d) min-max shape

To distinguish the general case from an AABB, an arbitrary bounding box is sometimes called an oriented bounding box (OBB). AABBs are much simpler to test for intersection than OBBs, but have the disadvantage that when the model is rotated they cannot be simply rotated with it, but need to be recomputed.

The **convex hull** (Avis & Bremner, 1995; Umbaugh, 2005) is the minimum area convex polygon that includes all the pixels of the layout element. In computational geometry, numerous algorithms are proposed for computing the convex hull of a finite set of points, with various computational complexities.

The "**Min-Max Shape**" for a set of points is defined as being the polygon obtained by the intersection of "Min-Max-X Shape" and "Min-Max-Y Shape" polygons, where:

- "Min-Max-X Shape" is obtained by joining successive minimum and respectively maximum X-coordinate points from all consecutive Y-constant scan-lines that may run across the input set of points.
- "Min-Max-Y Shape" is obtained in the same manner, thinking that the X-coordinated are temporarily exchanged with the Y-coordinates.

Please note that if the "Min-Max-X Shape" or the "Min-Max-Y Shape" is not well formed as polygons (this may occur when the input set of points is loosely connected) then the "Min-Max Shape" in itself cannot be defined (and eventually computed for further usage in the algorithm). An example of the mentioned shapes is illustrated in Fig. 2.

The "**Beta-Star-Shape**" (Boiangiu et al., 2008) for an input set of points is a bounding shape that tries to approximate as tight as possible the set. It is computed in a relatively time-consuming manner (an approximate $O\left(n^{3/2}\right)$ for the random case, where n is the number of points in the input dataset) by using successive refinement steps of the convex hull for that set of points and at every refinement step ensuring that all the points in the set are still contained inside the newly-obtained shape. Constructions details are presented in (Boiangiu et al., 2008). What is important is that the resulted "Beta-Star-Shape" is a valid polygon for any input point set, is tightly-wrapped (tightness may be controlled or traded vs. execution-time) around the points and is valid (contains all points inside no matter how loosely they are connected).

2.1 Algorithm

Shape 1	Shape 2	Priority
BR	BR	0
BR	OBB	1
BR	RS	2
BR	CH	3
BR	MMS	4
BR	BS	5
OBB	BR	6
OBB	OBB	7
OBB	RS	8
OBB	CH	9
OBB	MMS	10
OBB	BS	11
RS	BR	12
RS	OBB	13
RS	RS	14
RS	CH	15
RS	MMS	16
RS	BS	17
CH	BR	18
CH	OBB	19
CH	RS	20
CH	CH	21
CH	MMS	22
CH	BS	23
MMS	BR	24
MMS	OBB	25
MMS	RS	26
MMS	CH	27
MMS	MMS	28
MMS	BS	29
BS	BR	30
BS	OBB	31
BS	RS	32
BS	CH	33
BS	MMS	34
BS	BS	35

Tab. 1. BR – Bounding Rectangle, OBB – Oriented Bounding Box, RS – Rectangular Shape, CH – Convex Hull, MMS – Min-Max Shape, BS – Beta-Star-Shape

The first step of the algorithm is to assign priorities to pairs of shapes (Table 1).

The next step is to iterate every pair of layout elements from the page. For every such pair, the algorithm goes though the shape pairs (Table 1) in increasing order of priority and checks for intersections between the two bounding shapes for each element. For example, if the element pair contains 2 paragraphs, the algorithm will first compute the **bounding rectangle** shapes for each paragraph and intersect them. If there is an intersection, the next shape pair is used. This goes on until the bounding shapes do not intersect.

This process will happen for every pair of layout elements that are sent as input to the algorithm. The last shapes used for each layout element are the bounding shapes returned by the algorithm.

There is also a third, optional, step. Given the bounding shapes returned in step 2, the algorithm tries to smooth out complex shapes like BS by applying these transformations:

i) Removal of points until the shape reaches points lying on the CH (Convex Hull);

ii) The reconstruction of corners so that the shape resembles a BR (Bounding Rectangle) or RS (Rectangular Shape);

3 RESULTS

This algorithm was tested on a collection of scanned newspaper pages that contained large quantities of text as well as occasional images, headlines, tables, etc. Even though the complexity of the algorithm might be theoretically high in terms of the number of elements, if the input image contains many text paragraphs, the process will end much sooner because the paragraphs can often be enclosed in rectangular shapes easily. An example of how the algorithm performs is presented in Fig. 4.

Fig. 4 shows the shapes returned by this method on a typical newspaper fragment. The constructed shapes are:

- 1, 3, 4, 5, 10: Bounding Rectangles;
- 2: OBB;
- 6, 8, 9: Rectangular-Shape with more than one rectangle corner reconstructed (in a post-processing phase, result is up to the Bounding Rectangle);
- 7: Beta-Shape simplified in a post-processing phase by successive point removal until one intersection with other shape(s) occurred

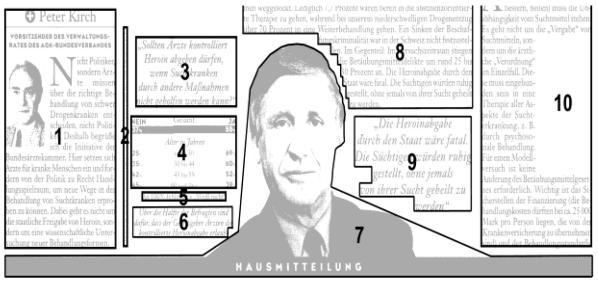

Fig. 4. Example of processed image

4 CONCLUSIONS AND FUTURE WORK

The algorithm/method presented in this paper is a simple and easy to implement solution to the problem of clipping fragments from a document to allow easy manipulation of the layout and structure. The main field of application is the content conversion domain. However, the algorithm can find other uses in such related fields as automatic image processing, or machine object identification.

There are two directions of further research concerning this algorithm. One involves the aim to enrich the algorithm with other shapes. This will add more accuracy and stability to the process. The new shapes have to be easy to test for intersection, thus they need to be as simple as possible. Another direction is in regards to the performance of the algorithm. Its running time may be improved by observations discovered empirically or deduced.

5. REFERENCES

Avis, D. & Bremner, D. (1995). How good are convex hull algorithms?, *Proceedings 11th A.C.M. Symposium on Computational Geometry*, pp. 20-28, ISSN 0925-7721, Amsterdam, June 1995, Elsevier Science

Boiangiu, C. A.; Bucur I. & Dvornic, A.I. (2008). The Beta-Star Shape Algorithm for Document Clipping, *Annals of DAAAM for 2008 & Proceedings of the 19th International DAAAM Symposium*, pp. 0127–0128, ISSN 1726-9679, October 22-25, 2008, DAAAM

Edelsbrunner, H. (1987). *Algorithms in Combinatorial Geometry*, Springer-Verlag, ISBN: 978-3-540-13722-1, Berlin

Pratt, W. K. (2002). *Digital Image Processing*, John Wiley & Sons, ISBN: 0-471-85766-1

Umbaugh, S. E. (2005). *Digital Image Analysis and Processing*, CRC, ISBN: 0849329191