ABSTRACT

Existing Computational Geometry algorithms aren't able to create a tight-fitting contour around document elements. Some of them don't follow the data points close enough, generating overlapping elements, while others, trying to keep close to the contour, break one element into pieces, falling between white spaces. The presented method both follows the contour closely and generates a single shape for a single element. The generation of the shape can be stopped at any time, either when there are no more intersections between layout elements or after a certain time elapsed, the result being valid at any given moment. In association with other algorithms it can offer a fast and clean solution for the problem of finding the non-overlapping areas resulted from the Layout Analysis document processing phase.

Keywords: layout analysis, alpha shape, bounding volumes, contour retrieval, automatic content conversion, scanned images

1. INTRODUCTION

Because of the exponential growth rate of the quantity of information, but also because of the aid computers give in document processing, document digitization has become a priority. Text is not the only information available in a document, but images, colors and layout now offer much more extra information. That is why OCR alone can't preserve the entire document, but there is also a need for preserving the look and feel. This is not straightforward neither for old documents which contain many types of nonstandard layouts and ornaments meant to beautify the page, nor for today's multi-column, multi-title and filled with pictures newspapers.

A document Layout analyzer has the role of enriching the OCR with the possibility of maintaining the aspect of the scanned document (fonts, header positions, paragraph characteristics, text and image positioning, and so on). One of the important aspects of a document layout analyzer is to separate headlines, paragraphs, images, etc. To do this...
correctly, it has to surround each element in a way that the generated groups don't overlap [22].

2. METHODS

The methods vary from fast, simple algorithms which encapsulate elements in axis aligned bounding rectangles to complex ones which manage to follow the text or images enclosed closely [13].

The simplest algorithm finds the extreme points on the X and Y coordinates of an element in the image and uses them to generate their Bounding Rectangle or Axis Aligned Bounding Box (AABB) [1] [4].

There are a number of problems that arise here:

- the layout has to be strictly rectangular without inserted elements through text
- even small skew angles can change the boxes completely

The second problem is resolved by using Oriented Bounding Boxes (OBB), which are the minimum area rectangles, of any orientation, that include the layout element. The results don't compensate usually for the complexity [4], so they are seldom used in this domain.

The convex hull encapsulates the element in a minimum area convex polygon [11] [14] [16]. This was a broad studied problem so now we know many computational geometry algorithms that can calculate it with \( O(N \cdot \log(N)) \) complexity. Still, as mentioned in the introduction, the layouts are usually not so clean, containing, for example, images between columns, generating concave text paragraphs.

The Min-Max Shape algorithm [22] generates a kind of concave polygon by intersecting the results from Min-Max-X and Min-Max-Y. Min-Max-X Shape is obtained by taking the minimum and maximum X coordinates on every Y raster line. The same for Min-Max-Y but with Y axis parallel raster lines and using the min and max Y coordinates. This could be enough for most layouts, but there are exceptions. Some really odd layouts from magazines old manuscripts can contain text areas that generate a C like shape that cannot be deduced with this method.

The Alpha-Shape [19] is a generalization of the convex hull and a sub-graph of the Delaunay triangulation [12]. It can be visualized as the contour created by a sphere or radius R that is rolling onto the exterior of the points in the same direction. It can’t go over the points. The sphere will enter between points that are farther away than 2R but will roll over the ones that are closer than 2R. The result is a tight concave polygon that contains all the points that were touched by the sphere. The problem with the construction of the Alpha-Shape is that it is usually based on the Delaunay triangulation, which is not very economical. Also, there is the necessity of choosiness how fine or coarse the generated polygon is, based upon how far apart are the farthest apart two neighboring points, so the sphere doesn't fall between them.
3. **BETA-SHAPE**

Even if Alpha-Shapes seem promising, they are unusable in the case of documents, because the distance between items, like text characters or lines inside paragraphs, is not controllable, so a radius that both generates a fine contour and doesn't split an object can possibly not exist.

That is why the algorithm described in this paper always creates only one tight concave shape [18] that contains all the input points.

The algorithm starts by generating the convex hull of a set of points using one of the known and robust computational geometry algorithms. Then it starts to iteratively refine each segment of the shape by adding new points until the distance between two adjacent points is lower than a distance given as input parameter. That segment will finally be integrated into the Beta-Shape [23] [24] and will suffer no more refinements. This distance can be considered as the input radius from the Alpha-Shape.

The points that don't make up the wrapper polygon are the candidate points. For every segment of the shape that is longer than an *(EdgeLength)* threshold distance, the algorithm searches for the closest point to the segment, from the candidates, firstly inside a search area near the segment. The search area is a square region that is inside the polygon and has one edge the segment that has to be split. This search area is used in order not to encourage the usage of potentially better candidates from other edges but also to restrict the search to a subset of the input points.

![Input points cloud](image1.png)

![Start from the Convex-Hull](image2.png)

![Edge search area and candidates](image3.png)

![In process of refinement](image4.png)

*Figure 1. Beta-Shape stages*
The initial Alpha-Shape contains all the input points inside. Because at every iteration only the closest point to each segment is inserted, there is no risk of leaving out any point or to generate self intersections. As a result, after any iteration the shape is fully valid. This means that the generation process can be stopped at any time, giving the possibility to trade between execution time and refinement quality.

As optimization, the input points can be stored inside a two-dimensional hash table, testing only the points from that hash table that contain the square search area:

<table>
<thead>
<tr>
<th>Bounding Rectangle</th>
<th>Relation between input points: $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$x_1 \leq x_2$</td>
</tr>
<tr>
<td>$X_{Min}$</td>
<td>$x_1 - y_2 + y_1$</td>
</tr>
<tr>
<td>$X_{Max}$</td>
<td>$x_2$</td>
</tr>
<tr>
<td>$Y_{Min}$</td>
<td>$y_1$</td>
</tr>
<tr>
<td>$Y_{Max}$</td>
<td>$y_2 + x_2 - x_1$</td>
</tr>
</tbody>
</table>

Table 1. The coordinates of the resulted bounding rectangle used as input for the two-dimensional hash-table

The condition that a point needs to satisfy in order to be inside the search rectangle is to be delimited by the four lines that construct the square. The Beta-Shape segment:

$$ax + by + c = 0$$

(1)

passing through the 2 points $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$ giving:

$$a = y_2 - y_1$$

$$b = x_1 - x_2$$

$$c = -ax_1 + by_1$$

(2)

The two perpendicular edges are:

$$a_1 = b \quad \quad a_2 = b$$

$$b_1 = -a \quad \quad b_2 = -a$$

$$c_1 = -bx_1 + ay_1 \quad \quad c_2 = -bx_2 + ay_2$$

(3)

The edge parallel with the segment at distance $EdgeLength$: 
\[ a_3 = a \]
\[ b_3 = b \]
\[ c_3 = c + EdgeLength \]  

(4)

Giving the constraints:

\[ b \cdot x - a \cdot y + c_1 \geq 0 \]
\[ b \cdot x - a \cdot y + c_2 \leq 0 \]
\[ a \cdot x + b \cdot y + c \leq EdgeLength \]
\[ a \cdot x + b \cdot y + c \geq 0 \]  

(5)

4. RESULTS

The images presented are a typical case of a justified text column, with uneven spaces between words and shorter and longer lines of text. Setting a small enough radius that enables the Alpha-Shape to be fine enough to separate the lines of text breaks the input points into clusters. The Beta-Shape with the same distance keeps the result connected and manages to cut out the white space after the shorter lines of text.

![Figure 2. Alpha-Shape versus Beta-Shape](image)

In the following test, filtered points from the player and the small image on the right are the input elements to the algorithm.

![Figure 3. Beta shape results in a “real-world” application](image)
In this test, all the methods presented in the introduction are used, depending on the type of elements. Area 7 is the one obtained with the Beta-Shape and the algorithm kept just the points needed to stop the elements from overlapping. Segments that don't generate intersections with other layout elements don't need to be split. If more than one Beta-Shape needs to be generated for a single document, they should be done in parallel so this optimization step can occur directly at the generation stage.

Tests show that the algorithm runs in $O(n^2)$ for random points but in about $O(n^{3.5})$ for real test images [21].

5. CONCLUSIONS

The Beta-Shape is a concept that may be successfully integrated in the Layout Analysis phase of a content conversion system because it offers a robust solution for separating document objects. It also offers valid results at every iteration step, thus being able to trade between quality and speed.

In association with other shape fitting algorithms it can be used for fast separation of the layout elements in order to accurately reconstruct the look and feel after a digitization process.
6. ACKNOWLEDGEMENTS

The work presented in this paper was funded by the Sectorial Operational Programme Human Resources Development 2007-2013 of the Romanian Ministry of Labour, Family and Social Protection through the financial agreement POSDRU/89/1.5/S/62557.

REFERENCES

1. Douglas R. Caldwell: Unlocking the Mysteries of the Bounding Box
2. Ross S. Swick and Kenneth W. Knowles: Geographic Database Search Interfaces and the Equatorial Cylindrical Equidistant Projection
4. Fast and Tight Fitting Bounding Spheres


ANNEX 1 – COMPLEX DOCUMENT LAYOUTS AND THE RESULTED “NON-OVERLAPPED” SHAPES
DAILY NEWS BRIEFING
Romney: in Israel, on foreign policy

Developing countries go mobile

Eine Stadt bangt um einen Arzt, der keiner ist
Keine Angst vor großem Tempo!