

Modern Approaches in Detection of Page Separators for Image Clustering

Costin-Anton Boiangiu, Dan-Cristian Cananau, Andrei-Cristian Spataru
Computer Science Department

“Politehnica” University of Bucharest
Splaiul Independentei 313, Bucharest
ROMANIA

Costin@cs.pub.ro, Dan_Cananau@yahoo.com, Andrei.Spataru@yahoo.com

Abstract: - This paper describes a model for detecting all types of separators on a document page and combining the results towards obtaining elements clusters on every document image page. The separators are determined by using various methods as for example the Delaunay triangulation. The physical layouts of the documents are always hard to extract, but determining the simplest separators found in most documents is the starting point for correct layout detection.

Key-Words: - automatic content conversion, Delaunay triangulation, separator detection, line detection, contour points, neighborhood relations, white-space detection, page clustering

1 Introduction

The new available technologies for scanning and printing have brought up the need and desire for better document processing. Even though the current software solutions provide reliable conversion, the next step is to expand the electronic understanding of the document to recognize the logical structure (chapter delimitation and titles, sections, headings, paragraphs, authors and affiliation, annotation, footnotes, references, commentaries, related pictures and schemes, page number).

The current paper goal is to provide a reliable algorithm that may improve the detection of the logical structure and furthermore content conversion. And as almost everything can be decomposed in basic elements, so can the documents. And for this reason a combined algorithm for detecting separators and the areas delimited by them in the document page is presented.

The separators can be roughly divided into categories depending on their shape or geometrical characteristics. The classification is the following:

- Line separators;
- Line-based separators;
- White space separators;
- Arbitrary-form separators.

Most of the approaches used for line separators detection are based on the idea that lines are entities with either width or height greater than the other and some constraints about the ratio between width and height. More evolved approaches also take into consideration the angle of the line in order to connect

broken lines or dotted lines. However this type of approaches is geometrically dependent and can detect only line separators. There are also some algorithms that evolved from the basic idea of distance and use mathematics as a solution. Examples of line detection algorithms can be found in [11], [12].

Most of the white-space detection algorithms are based on the size of the collection of white pixels on an axis with respect to the other axis of the Cartesian system. This type of approaches is size and orientation dependent, just like the line separator detection algorithms, but has a better degree of certainty. However, a more complex and reliable method is required for this type of detection.

This paper presents several approaches towards determining different types of separators and combining the result in order to obtain a hierarchy of the document based on detected areas.

The line detection algorithm used [2] is one based on a non-standard parameterization in the Hough transform.

The white-spaces are determined using a simple algorithm based on resampling, which provides the desired output in most of the cases.

For the arbitrary-form separators, an approach based on the information obtained by applying the constraint Delaunay triangulation algorithm is used. This approach is introduced in order to overcome the angle orientation problems of the lines and the difference in width or height. Also it will not be dependent on the entity geometry, which will allow us to detect curved or arbitrarily-shaped separators and also image or table borders as separators.

The outcome of the algorithm is the already mentioned clustering which provides a separation of the document areas based on the results obtained at the previous steps. In this way, a starting point for the recognition of the logical structure is created.

The novelty of the presented approach is that it is a “top-down” one and not a “bottom-up” one as many other algorithms used for document page clustering. What is meant by the previous comparison is that this algorithm has the purpose of breaking the page in collections instead of trying to group different recognized objects (using some heuristically-obtained measurements) into corresponding categories thus performing a first-step clustering. The use of the Delaunay triangulation brings a mathematical approach towards obtaining a component interconnection structure. The Delaunay triangulation method maps extremely well on this task because it tends to cluster elements in a first step based on the neighborhood relations and also based on the existence of a direct “line-of-sight” between the directly-connected elements. This simulates very well the natural tendency of the human eye which intuitively “connects” elements based on the same characteristics.

2 Input Creation

Before starting the actual algorithms, a preprocessing of the input has to be made.



Fig. 1: A black and white conversion of an initial grayscale image.

First of all, the document is in an electronic format and in order for the following steps to function properly the image has to be black and white. To achieve this goal a simple black and white conversion algorithm is used. It is not the purpose of this paper to

present such algorithms, but only to use the results previously obtained. For further reading refer to [1]. A black and white conversion of a grayscale image is presented in Fig. 1.

The next step is the actual input selection which implies generating the entities in the image. An entity is a collection of connected black pixels which are determined by a simple algorithm that starts from such a pixel and goes through all of the black pixels connected directly or indirectly to the starting point. In the end, a collection will be obtained and the process is repeated until all collections are found. In Fig. 2 all the entities found are encapsulated in their bounding rectangle.

After this preprocessing has been done, the actual algorithm can begin. The following steps are used for detecting all categories of separators and combining the results in the final stage.

Sitteſt in dread Diſcourſe, or Fello
Who joy to ſee the Honour of the
Or whether, mounted on Cherubic
Thy ſwift Career is with the whirl

Fig. 2

3 How to find the arbitrary-form separators

In order to find the arbitrary-form separators in a document it is needed to have a method that is independent of concepts as size and line-like shape. The idea is the following: in a page, if each character, image or collection of pixels is seen as an entity, each of them can be connected with the neighboring entity by using Delaunay triangulation. In this way, the obtained results are filtered and only triangles that connect two entities by respecting the Delaunay algorithm are considered.

Previous Delaunay methods are used for detecting various elements of structure as for example the title and author regions in [6].

Most of the characters will be connected to up to eight entities which are the two entities on the same line, three above and three below, and so if the ratio between the connected points on the current entity (further referred as “*current points*”) and the connected points on all other entities connected by triangles (further referred as “*destination points*”) is computed it will give a relatively high result, around the value one.

However this rule generally does not apply to separators. Because they extend on a few rows, they

are connected to far more entities and have far more connectable Delaunay points, the number of connected points on the separator will be significantly greater than the number of points on the other entities connected by triangles with the separator. First it is needed to determine the input for this type of detection and check for eventual problems that may occur.

3.1 Input selection

The input will be selected by obtaining all the collections of connected black pixels as stated in the preprocessing stage. This is not enough for passing to the next step because of the need to apply constrained Delaunay triangulation on a set of vertices. These are obtained by using a simple contour detection algorithm on the input data (for other contour detection algorithms see [5]).

Von Rolf Dietrich Schwartz

Die Bundesregierung sagt voraus, daß es 1998 ein Wirtschaftswachstum von bis zu drei Prozent und am Jahresende weniger Arbeitslose geben wird. So steht es im Jahreswirtschaftsbericht, den das Kabinett am Mittwoch verabschiedete. Die Opposition sprach von „Schönfärberei“, Gewerkschaften vom „Prinzip Hoffnung“.

BONN, 11. März. Die Bundesregierung rechnet in ihrem Jahreswirtschaftsbericht damit, daß bis zum Jahresende

Fig. 3: Initial image.

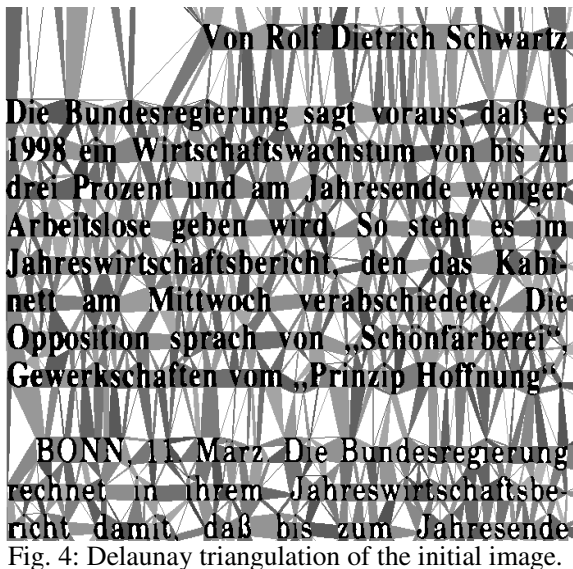


Fig. 4: Delaunay triangulation of the initial image.

3.2 Delaunay algorithm

The next step is to apply the Delaunay triangulation (as in [9], [10]) on the obtained input and filter the triangles in order to obtain only the ones that connect two entities. Next, the already mentioned ratio between the *current points* and *destination points* is computed. The final result will be a number of ratios from which it will be easy to select the separators based on their properties.

As it can be observed in Fig. 3 from the initial picture, the pass to the next step is done by applying the contour detection algorithm. The final image processing step is the Delaunay triangulation (shown in Fig. 4) where each distinct gray shade illustrates the collection of all the triangles that connect the same pair of entities. In fact there are lots of thin triangles connecting all the points between two entities, but in the final result which is seen in the figure, all the triangles between the same two entities are combined into one bigger “proximity”.

4 How to use the neighborhood for arbitrary-form separators detection

As already stated, first of all, the selection of entities is based on a simple algorithm of introducing all connected pixels different from the background in the corresponding entity. A connected pixel is referred as a pixel neighboring another pixel different from the background. Now that the input is selected, the algorithm has to create the vertexes for the Delaunay triangulation and these will be selected from the set of contour points of each entity.

4.1 Contour generation algorithm

Every entity can be seen as a collection of horizontal segments of pixels. The contour of the entity is given by the points on the outside border of the entity and in some cases the points on the inside border when the inside border exists like in Fig. 5.



Fig. 5: Result of the contour detection algorithm.

These contour points are in fact the extremities of the horizontal and vertical run-length segments of the entity. The algorithm for finding these is based on a simple idea of determining the connected pixels on a horizontal axis this time and taking the extremities of the found lines. To be more specific, it passes through each line and finds the starting pixel and the ending pixel.

However the entire contour is not needed because it is desired to select as input only the points that give (for the triangulation) a direct arc between two different entities (inter-entity arc) and not between points of the same entity (intra-entity arc) or between three distinct entities.

What this means is that if the entity does not intersect others inside its bounding rectangle, the bounding rectangle of this is taken and the points that cannot be seen directly from the exterior using axis-aligned observation lines are removed from the selection as in Fig. 6.

Also, if an entity contains a hole and the hole does not contain other entities, the inner contour of the hole is not generated. In order to point out the steps of the algorithm, the images below are presented.



Fig. 6: Another result of the contour detection algorithm using axis-aligned observation removal.

4.2 Constrained Delaunay triangulation

The constrained Delaunay triangulation is used for generating a mesh between a set of vertexes by following a simple algorithm: three points are united so that the circumscribing circle of every triangle is empty.

An example of constrained Delaunay transformation of the initial image in which the triangles are created only between two entities and the minimal arcs are selected can be seen in Fig. 7. The purpose of this paper is to use the results of such an algorithm for separator detection. Further information on the Delaunay triangulation is presented in several books and papers as [7], [8].

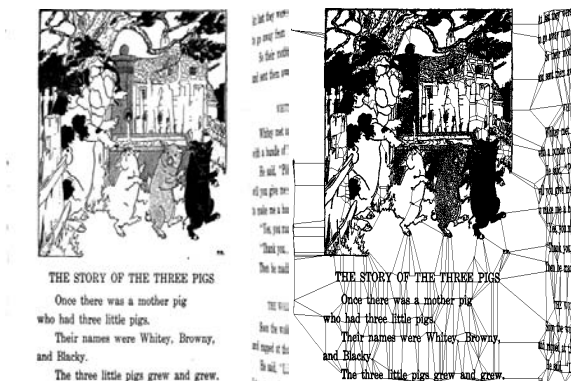


Fig. 7: Applied Delaunay triangulation and filtering of the minimal arcs between two entities.

After applying Delaunay, all the contour points found at the previous step will be joined in triangles that respect the circumscribing circle property. The results of applied Delaunay on a document, which contains separators, can be seen in Fig 8.



Fig. 8

4.3 Main algorithm

The purpose of this stage is to compute the already mentioned ratio of *current* and *destination points*. First of all, to find these points the triangles are filtered in order to keep only the ones connecting two entities. This is done by checking each of the three points for the belonging entity. If there are only two different entities connected by the three points then the triangle is accepted, otherwise the triangle is removed.

Each entity is taken and counted for the *current* and *destination points* and then the ratio between them is computed. This initial result will give a starting step for the algorithm. Some results can be observed in the following table for the document in Fig. 10:





Entity	Ratio
	2.7328
	3.2832
	0.7610
	0.9662

Table 1

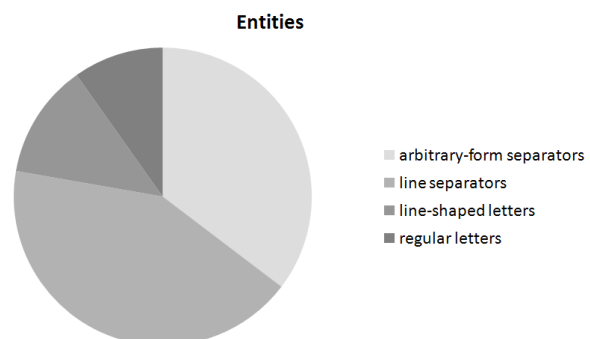


Fig. 9: Pie chart view of the average results obtained by the algorithm on a set of images.

As it can be seen from the results, the algorithm is correct and there is a difference between separators and letters, but it may be too small to be useful when making a decision for entities classification.

However, these results show an important fact: the difference is real and it can be enhanced by multiplying it with another measure that can make the difference between separators.



Fig. 10: Input image for the algorithm.

4.4 First improvement

The next step to enhance the obtained ratio is to multiply it with the area of the entity.

By doing so, the value of the results will be far greater for separators like the first one presented in Table 1, which are more similar to pictures or symbols than to lines.


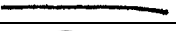


Entity	Ratio
	2102128
	309269
	6810
	5293

Table 2

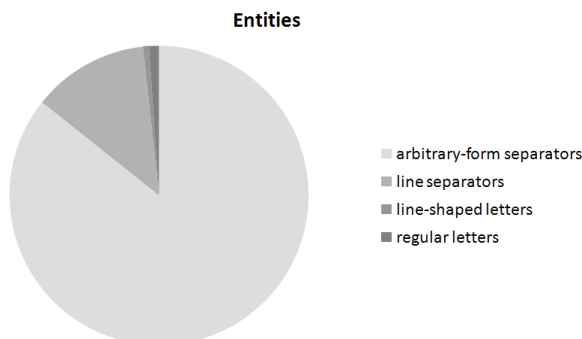


Fig. 11: Pie chart view of the average results obtained by the first improvement of the algorithm the same set of images used for Fig. 9.

Also the text boxes and the image borders will be detected with a higher ratio than all other elements. Table 2 provides the results obtained for the same input as for the first run of the algorithm.

Even though the differences between the line and characters are still small with respect to the drawing, the algorithm shows a great difference for the symbol separator entity.

4.5 Second improvement

The last approach designed to enhance the algorithm results is oriented towards the actual line separator detection. It is known that a line separator is mainly made out of colored pixels different from the background color.

The idea of line is an entity oriented on an axis filled with pixels of the same color. So, in order to emphasize this characteristic, the convex hull of the entity is created and then its convex hull fill ratio is computed. The convex hull fill ratio is the number of pixels of the entity divided to the entity convex hull surface area. By doing the previously mentioned operations, the lines will have the convex hull fill ratio close to 1, with some small differences given by scanning errors or printing errors. And so the results obtained for the same input as for the previous two cases can be observed in Table 3.





Entity	Ratio
	0.884348
	1.680385
	0.6445
	0.6916

Table 3

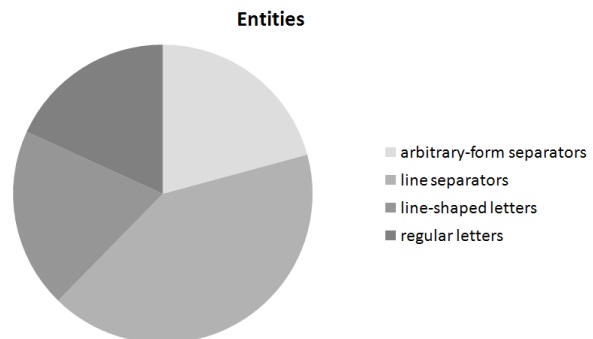


Fig. 12: Pie chart view of the average results obtained by the second improvement of the algorithm the same set of images used for Fig. 9.

It can be easily seen that the method favors line detection and the results prove that the algorithm works correct.

4.6 Algorithm interpretation

The algorithm has three versions depending on the necessity to find arbitrary-form separators that do not respect a linear rule or to discover the lines as a lot of other algorithms that use different approaches.

But the result will be an array of ratios of the entities on the page. This is not enough because it needs to determine the separators with a high degree of certainty. If the algorithm is simply used without

any enhancements then some of the entities which are letters or punctuation marks will have ratios similar to separators and a decision cannot be made with enough certainty. That is why the algorithm has to be used with both enhancements, each of them being a checking for the other.

So, the separators that do not respect a linear rule will have a very high ratio and a threshold can be used in order to determine them. A safe threshold can be above 75% of the difference between the maximum and minimum values obtained when calculating the ratios. However, this selection can be enforced by checking the values of the measurements when applying the convex hull multiplication enhancement. The separators tend to have also a value around average in this version.

The same idea can be applied to the line detection enhancement, by using the convex hull fill ratio and then checking the results with the area multiplication of the ratio.

Even though the algorithm is designed for both line and arbitrary-form separator detection, in this paper it will be used only for the second type because it has a higher degree of reliability and there are line-detection algorithms that provide better results.

5 How to find the line separators

One of the algorithms that provide better results than the previous one is presented in [2] and uses a non-standard parameterization of the Hough transform, called "Discrete Parameterization". This is also the method used towards obtaining the results in this paper. An example of the applied algorithm can be seen in Fig. 13. The line detection phase will consume all the line segments (in case of broken, dotted lines) so that, for a better precision of detection, all these segments may be removed from the input of the presented algorithm.

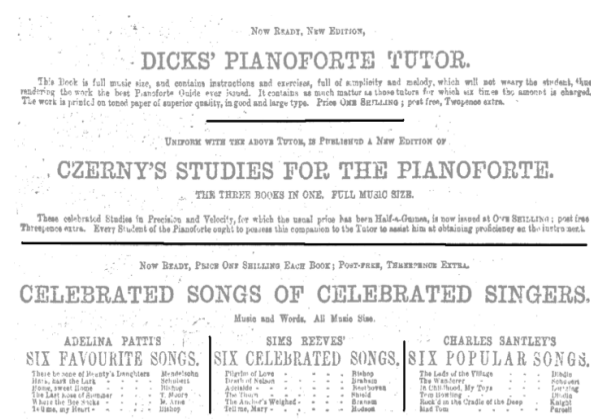


Fig. 13: Result of the line separators detection algorithm on an image, where the lines have been drawn in bold.

6 How to find the white-space separators

The last type of separator that needs detection is the white-space that splits several text areas, pictures or tables.

The main purpose is to determine these separators, but without including the white spaces between the text lines, known as line spacing.

In order to obtain this result, a simple resampling algorithm is used to blur the image and retrieve the white spaces. The purpose of the blur is to eliminate from the possible output the line spacing which will contain pixels of other colors than white in the final image.

6.1 First step - Downsampling

The first step is to take the initial image and downsample it using a triangle filter (Fig. 15) with a fixed very small window (best results were obtained for a value of around 4K pixels).

The downsampling is done in the following way: a triangle filter is moved from pixel to pixel and the centered pixel value is set to the value of the neighboring pixels multiplied by the filter window.



Fig. 14: The page after resampling and resizing it with the triangle filter of fixed window.

The triangle filter window starts from the edges with the null value and increases to the centre when it reaches the maximum value. Afterwards each value of the window is multiplied with the values of the corresponding pixel regarding the position of the center.

The technique is similar to the Gaussian blur which also has the purpose of enhancing the middle pixel and by doing so smoothing the image. The obtained result seems like a blurred version of the initial image at a very small dimension (Fig. 14).

The mathematical equations used for the computation of the triangle filter are given below:

$$T(t) = \begin{cases} 1-|t|, & |t| < 1 \\ 0, & \text{otherwise} \end{cases}$$

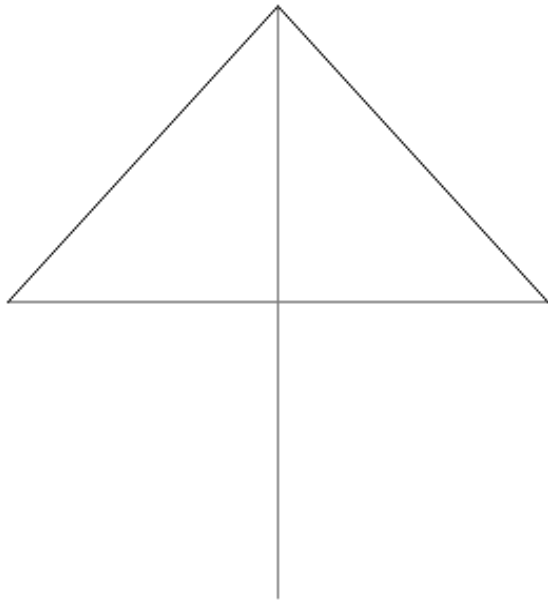


Fig.15: A graphical representation of the triangle filter

6.2 Upsampling

Next, the image is upsampled by using a B-spline filter (Fig. 17).

The idea of resampling is to create a filtered version of the initial image that can also vary in size from the input. In this case, the picture was first downsampled to 4K pixels and afterwards it was upsampled to the initial size. By doing this operation, the size of the picture was maintained, but the content inside has been changed, the new image being a blurred version of the initial one because of the two filtering operations (Fig. 16).

The B-Spline filter is similar to the triangle filter, but better results are obtained for upsampling by using this type of filter. The mathematical equations are given below:

$$B(t) = \begin{cases} \frac{1}{2}t^3 - t^2 + \frac{2}{3}, & |t| < 1 \\ \frac{1}{6}*(2-t)^3, & 1 < |t| < 2 \\ 0, & \text{otherwise} \end{cases}$$

The difference between the small version of the image after applying the downsampling and

reconstructed image created by reversing the algorithm is noticeable. Even though both of them seem to be a blurred image of the initial, the first one is actually a compact version. At a closer look on Fig. 14 the pixels can be seen entirely and their colors can be noticed without any zooming effects because of the small size and color pattern (see Fig. 18).



Fig. 16: The page resampled with the same filter to the initial size. The blur effect is visible (the reconstructed image).

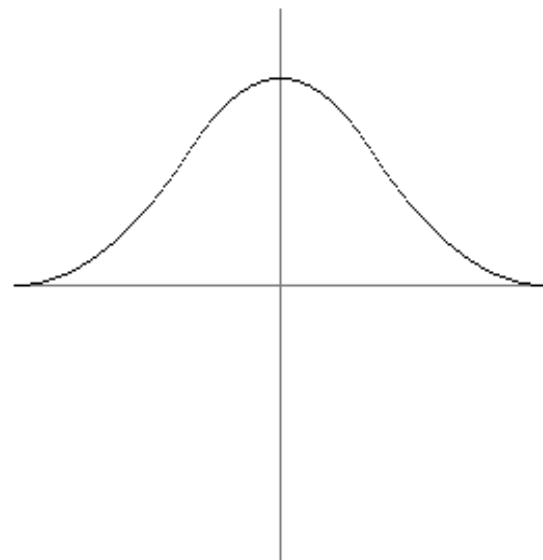


Fig. 17: A Graphical representation of the B-Spline filter

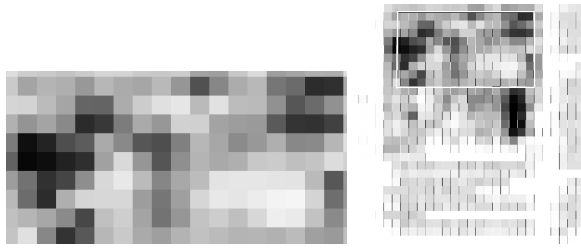


Fig. 18: A closer look on the resampled picture.

The fact that the second image, Fig. 16, is blurred is a result of successively applying the filter. A simple transformation of the initial image (see: [13], [14] for image processing) would not have given such a good effect and zones that should have been united would have been left singular.

6.3 Black and white conversion

The last step is to apply a black and white conversion on the blurred image based on a simple centre-thresholding technique and so the result from Fig. 19 is obtained. The purpose is to mark as white the areas of white-space separators in the image.



Fig. 19: A simple black and white transformation applied to the reconstructed image.

6.4 Correct zone detection

In the black and white image, the possible join probabilities of different zones are emphasized and an overview of the actual structure can be seen.

The effect of the resampling before the actual conversion is to emphasize the white spaces with

respect to the text or other elements. This approach favors the large spaces that have a continuous distribution in the page.

This last concept of distribution refers to the spaces that are not bounded or have very large dimensions on both axis between the bounds. As an example, in the obtained black and white image it can be clearly observed that text from the right is not in the same zone as the text from the middle or the picture and so, a difference is determined (Fig. 20).

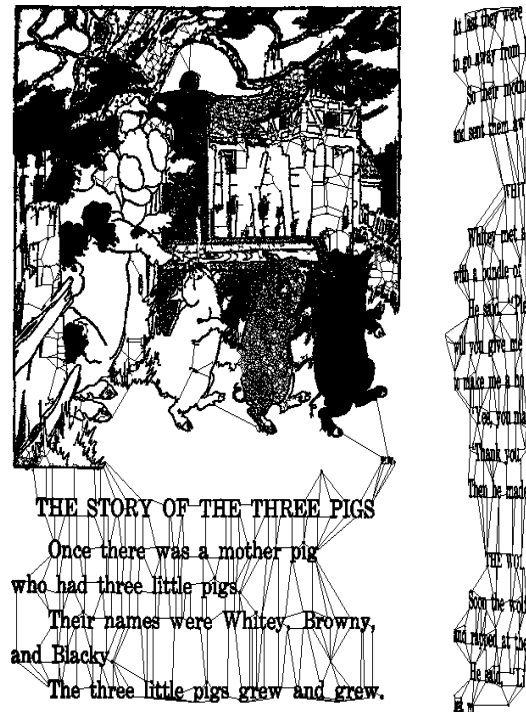


Fig. 20: The zones are the parts which have all the elements connected by Delaunay triangles.

7 Final Processing

In the previous sections of the paper several steps towards determining the separators on a document page have been presented. The final goal is to determine the element clustering (see also [3] for other clustering algorithms) of the page and, by doing so, to create an overview of the layout structure.

The order in which the results of the separator detection are used is not important as long as the first step is the determination of the white spaces. As presented previously this type of separators are found and the Delaunay connections that cross them are eliminated.

However this processing is not enough because there are still Delaunay connections between separators and other areas which should be eliminated. As an example the picture in Fig. 20 is clearly different as an area from the texts on the

bottom and right sides.

So, the next step is to find the line separators and then the arbitrary-form separators with the presented algorithms. Afterwards all the Delaunay connections starting from the separator or passing through the separator are eliminated.

As it can be seen in Fig. 21, the two points above the characters "PIGS" have been detected as a line and the corresponding arcs have been removed. An example of a correct page clustering is presented in Fig. 22.



Fig. 21: The two points have been detected as a line separator.

As stated before, by using the Delaunay triangulation a structure of the interconnected components can be created. The interconnected components are pairs of two entities bounded together by the corresponding arcs from the Delaunay triangulation.

The clustering effect is obtained by validating or not these arcs. An arc is not validated if the algorithm decides that it cannot pass through the detected obstacles. The obstacles fall in our case in two major categories: the separators found on the page and the white-space areas. Both of them must "block" the arcs between different interconnected structures. The "blocking" operation is realized by: (for the first category) iterate through all the length of the arc and verify if any of the pixels encountered belong to a separator and (for the second category) superimposing the arcs onto a black/white mask image which has the purpose of invalidating the arcs that passes through white regions.



Fig. 22: After detecting the separators and removing the connections to them, the zones are found correct.

8 Conclusion

8.1 General considerations

The algorithm presents an innovative idea for separator detection, taking this type of approaches previously implemented a step further. The use of constraint Delaunay triangulation provides the means of exploring the dimensions and characteristics of the separators and especially lines, without having to worry about angle orientation or dimension ratio problems. It is a simple method and it proved itself effective for about 60% of tested data, the numbers of failures being mostly given by scanning or printing errors.

The presented method combines the results of a pure mathematical model, the triangulation, with the geometrical constraints between the elements of images that are object of automatic content conversion.

8.2 Separators

The white-space separator detection algorithm also uses an innovative idea towards obtaining the same processing for results. If in other approaches the blurring effect is created directly by using a Gaussian blur or other types of such algorithms, the approach provides a better effect by eliminating the line spacing from the output.

As presented, one of the most important things that have to be considered when trying to find the text zones in a document is finding the separators. Without a good separator detection algorithm, like the one presented in the paper, lines, images, tables and other strange-shaped entities in the page will be connected erroneously to text zones.

But the method provided above brings a solution to this problem by combining different separator detection approaches towards a final correct clustering of the document page. The advantage of the detection algorithm is that it uses a different method for each type of separators, making it more reliable and producing better results because it uses the optimal detection for each separator.

8.3 Clustering

The top-down approach is innovative for this type of clustering because of the property of breaking the page into components and not creating them from smaller entities. The Delaunay triangulation enforces this approach by bringing the mathematical knowledge required for the breaking process in the form of Delaunay arcs between two entities.

The final filtering of these arcs for separator crossing elimination results in a correct clustering of the initial document pages.

References:

- [1] Costin-Anton Boiangiu, Andrei-Iulian Dvornic, "Bitonal Image Creation for Automatic Content Conversion", *Proceedings of the 9th WSEAS International Conference on Automation and Information*, Bucharest, Romania, June 2008, pp. 454-459.
- [2] Costin-Anton Boiangiu, Bogdan Raducanu "Robust Line Detection Methods", *Proceedings of the 9th WSEAS International Conference on Automation and Information*, Bucharest, Romania, June 2008, pp. 464-467.
- [3] Moh'd Belal Al-Zoubi, Amjad Hudaib, Bashar Al-Shboul, "A Fast Fuzzy Clustering Algorithm", *Proceedings of the 6th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Data Bases*, Corfu Island, Greece, February 2007, pp. 28-32.
- [4] Costin-Anton Boiangiu, Dan-Cristian Cananau, Spataru Andrei, "Detection of Arbitrary-Form Separators Based on Filtered Delaunay Triangulation", *Proceedings of the 9th WSEAS International Conference on Automation and Information*, Bucharest, Romania, June 2008, pp. 442-445.
- [5] Juan Zapata, Ramon Ruiz, "A Hybrid Snake for Selective Contour Detection", *Proceedings of the 6th WSEAS International Conference on Signal Processing, Robotics and Automation*, Corfu Island, Greece, February 2007, pp. 230-235.
- [6] Yi Xiao, Hong Yan, "Location of title and author regions in document images based on the Delaunay triangulation", *Image and Vision Computing*, Volume 2, Number 4, April 2004
- [7] Jonathan Richard Shewchuck, "Constrained Delaunay Tetrahedralization, Bistellar Flips and Provably Good Boundary Recover", University of California at Berkeley Course Notes.
- [8] Jonathan Richard Shewchuck, "Delaunay refinement algorithms for triangular mesh generation", *Computational Geometry: Theory and Applications*, Volume 22, May 2002.
- [9] Steven Fortune, "Voronoi Diagrams and Delaunay triangulations", *Handbook of discrete and computational geometry*, CRC Press, 1997, pp. 377-388.
- [10] Jonathan Richard Shewchuck, "Tetrahedral Mesh Generation by Delaunay Refinement", *Proceedings of the Fourteenth Annual Symposium on Computational Geometry*, Association for Computing Machinery, Minneapolis, Minnesota, June 1998, pp. 86-95
- [11] Liu Wenyin, Dov Dori, "A protocol for performance evaluation of line detection algorithms", *Machine Vision And Applications*, Volume 9, Numbers 5-6, Springer Berlin / Heidelberg, March 1997, pp. 240-250.
- [12] D. S. Guru, B. H. Shekar, P. Nagabhushan, "A simple and robust line detection algorithm based on small eigenvalue analysis", *Pattern Recognition Letters*, Volume 25, Elsevier Science, 2004.
- [13] Steve Mann, "Intelligent Image Processing", John Wiley & Sons, 2002.
- [14] William K. Pratt, "Digital Image Processing", John Wiley & Sons, 2002.
- [15] Costin-Anton Boiangiu, "Elements of Virtual Reality", Macarie, 2002.
- [16] B. Chen, and L. He, "Fuzzy template matching for printing character inspection", *WSEAS Transactions on Circuits and Systems*, Issue 3, Vol. 3, 2004.
- [17] L. M. Sheikh, I. Hassan, N. Z. Sheikh, R. A. Bashir, S. A. Khan, and S. S. Khan, "An Adaptive Multi-Thresholding Technique for Binarization of Color Images", *WSEAS Transactions on Information Science and Applications*, Issue 8, Vol. 2, 2005.
- [18] M.I. Rajab., "Feature Extraction of Epiluminescence Microscopic Images by Iterative Segmentation Algorithm", *WSEAS Transactions on Information Science and Applications*, Issue 8, Vol. 2, 2005.