Beta-Shape Using Delaunay-Based Triangle Erosion

COSTIN-ANTON BOIANGIU, MIHAI ZAHARESCU
Department of Computer Science and Engineering
University "Politehnica" of Bucharest
Splaiul Independentei 313, Bucharest, 060042
ROMANIA
costin.boiangiu@cs.pub.ro, mihai.zaharescu@cti.pub.ro

Abstract: - The aim of this paper is to offer a faster and more rigorous way for calculating the Beta-Shape. The Beta-Shape is a tight-fitting concave polygonal contour, similar to the Alpha Shape, but which remains connected when data points are farther away than the desired resolution. The proposed method uses a computational geometry approach. We also propose an improvement to the original Beta Shape, aimed at generating a more natural flowing geometry. The complexity of the original algorithm are calculated and compared to the complexity of this algorithm.

Key-Words: - Beta Shape, Alpha Shape, Convex Hull, contour, triangulation, mesh generation, segmentation

1 Introduction

The original motivation for developing a tight fitting contour around a set of given data-points came when we needed to surround each element from a scanned newspaper page (paragraph, graphic, heading, etc.), so that they may be sent individually or in order to an OCR system. The encapsulating contour must not contain fragments from neighboring elements because the intersecting regions will be interpreted twice by the image to text conversion system (once for every element) and inserted throughout the text.

Newspapers have very nonstandard layouts, like random shaped images inserted between columns of text, with text flowing around them. This implies that a simple bounding box is not good enough, not even oriented bounding boxes or convex hulls are enough, as the text contains concavities from the elements inserted. Other methods that can be used are min-max shapes, alpha-shapes or conformal alpha shapes. Some of the drawbacks of each algorithm are mentioned here [2]. We will summarize them in section 2 and explain why not even the more advanced ones are appropriate for the task.

Stepping away from the layout analysis domain, an algorithm for generating a tight fitting, connected, concave polygon, around a set of given data points is useful in domains like surface generation, collision detection optimization, mesh simplification and others that rely on computational geometry approaches. When a 3D scanner scans a single object, the sampled 3D vertexes must form a single connected geometry. The state of the art conformal alpha shapes fail to keep the geometry connected for all resolutions, while the others fail to recreate the finer details (alpha shapes) or concavities (convex hulls).

The paper is structured as follows: Chapter 2 discusses the drawbacks of similar methods regarding scanned document segmentation; Chapter 3 describes the original proposed Beta-Shape algorithm; Chapter 4 deals with the complexity of the original algorithm; Chapter 5 proposes a new way for generating the beta shape and; Chapter 6 improves the output of the algorithm, generating a more naturally flowing mesh. At the end conclusions are drawn.

2 Drawbacks of Similar Methods

Here we discuss some known methods for encapsulating objects, mainly from the point of view of how well suited are they for our document splitting problem.

The simplest method uses axis aligned bounding boxes. They are simple to calculate and intersections can be found easily, but that is about all they can do. For example, they can't even handle small rotations (which are very present in scanned documents because of imprecise paper alignment). Even small rotation angles cause neighboring columns to intersect on one of the axis. In our spitting problem they can be used as a first stage intersection detection.

The OOBB approach resolves the rotation problem but still assumes that all features are rectangular.

The convex hull is the smallest convex set that contains all the data points. It can find more complex objects, but magazine and newspaper articles usually include graphics inserted between text, thus introducing concavities.

The min-max shape is formed by the intersection of two other shapes that can be calculated with little cost: the min-max-x shape and the min-max-y shape. They are generated by taking the minimal and maximal point from every raster line, parallel to each axis. This can handle insertion of rectangular objects, but “C”-like shaped objects can't be handled by this approach.

The alpha-shape is a filtration, or sub-graph, of the Delaunay triangulation and a generalization of the convex hull. Intuitively it is created by rolling a sphere around the input points; the contour left by the rolling ball is the alpha shape. The problem arises when the alpha parameter, the ball's radius, is small enough to fall between distant data points, effectively splitting the geometry. Increasing the radius is not a solution, as the ball won’t be able to detect small cavities in the shape, thus lowering the shapes resolution.

Conformal alpha shapes [3] are an optimization of the alpha shape algorithm, aimed at changing the radius parameter adaptively with the local resolution of the mesh. This algorithm is thus able to generate a shape from non-uniform sampled datasets. However, the author mentioned that there are cases when the algorithm fails and generates holes in the constructed mesh. This happens because it selects a subset from the simplices generated by the Delaunay triangulation.

The largest spheres that fit in the empty spaces between data points are called *maximal open balls* and the union of their centers is the *medial axis*. The distance from a data point to the medial axis is the respective point's *feature size* (how dense the mesh is in that region). In regions where the sampling abruptly changes (ex: on a side of a data point the feature size is very small and on a different side it is very large) the sphere around the data point is too small to generate a triangle to the distant neighbor, thus generating a hole.

The mentioned algorithms are not specially built for page segmentation but rather for problems like intersection detection; pattern recognition; image processing; statistical analysis; global information system; reconstructing objects from a cloud of read data points, etc.

3 Beta-Shape Algorithm

The Beta-Shape algorithm was first proposed by Boiangiu [1] and it was developed just for this purpose: to generate a connected border, without holes, around all data points and as close as possible to the points, with the minimum number of edges, irrespective of the sampling.

Broadly described, the algorithm starts by computing the convex hull and then refines this border by iteratively adding data points to all segments larger than a given radius. At each iteration, for each border segment larger than the desired radius, the algorithm picks the closest point to the respective edge from a square region. This region is defined as the square having an edge common to the border edge and oriented towards the center of the polygon. If an edge in the border is smaller than the radius it will remain unchanged until the entire process stops and will finally be part of the resultant hull.

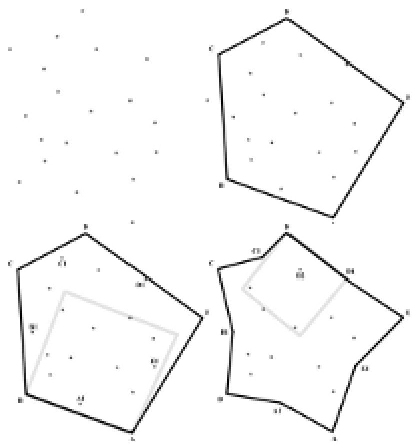

Fig.1 Steps of the old Beta-Shape algorithm

Using the square regions reduces the number of stolen best candidates from the neighboring regions and also the search domain for the best point. However, when the border segments form sharp angles, they could possibly ask for the same point as best candidate. This means that an additional verification is needed in order to avoid stealing a neighbor’s best candidate. This is also valuable for close-by edges that are inside each other's searching square.

In order to verify if a point is inside the square, four inequalities are inquired for answering the questions:

- is the point between the two perpendicular lines on the border segment?
- are they at the right depth?
- are they inside the shape?

It should be mentioned that a hash table can be used to fast-recover the search.

ISBN: 978-960-474-351-3

4 Complexity

This chapter discuses the complexity of the original Beta-Shape algorithm.

The author mentioned in the original paper that the overall complexity, deduced by experimentation, is $O\left(n^{\frac{5}{2}}\right)$ for random points and $O\left(n^{\frac{3}{2}}\right)$ for real data.

To calculate the complexity, we sketch the program's steps:

1. Calculate the convex hull in $O(n \log h)$ time; where h is the number of points on the hull
2. Run $\log(n-h)$ iterations (because at each step, for each edge we include one more point in the border, doubling the number of edges and points on the border).
3. At each iteration, for every edge, find the best candidate. In order to find the best candidate, test if the point belongs to a square region, in $O(n-h)$ time and find the best candidate from the c points inside the square in $O(c)$ time.

Thus, the total complexity is about:

$$O(n \log h) + O((n-h)c \log(n-h)) \qquad (1)$$

This explains the different complexity results for random and real data points: points in real binary images form clusters of white and black, giving a large h (number of points on the convex hull) for the complex hull. On the other hand, random numbers are spread, h being of $O\left(n^{\frac{1}{3}}\right)$ size, as calculated here [4].

Another attenuating factor for real images is that the edges are very small from the start. The data points are neighboring pixels - being the result of bitonal image conversion or wanted compact filtering of small white spaces - giving a small search region and small c.

5 Beta Shape Algorithm Improvement

In this chapter we propose a new method for obtaining the same results as the original Beta-Shape algorithm.

The proposed improved algorithm for calculating the beta-shape is still iterative. It erodes triangles one at a time from the Delaunay triangulation of the input points, verifying at each step the connectivity of the mesh.

The Delaunay triangulation generates the best connections that can be made between all data points in order to minimize triangles with sharp angles. The method avoids as much as possible the generation of skinny triangles. This is effectively achieved by testing that none of the points is inside the circumference formed by any other triangle.

Proof: if we have 4 points and a triangulation for them, the common edge of the two triangles formed is called *illegal*, if by flipping it we obtain a larger minimum angle. From Thales' theorem it can be noticed that if three of the points form a circle and the forth point is inside the circle, the value of the smallest corner will be smaller than that of the smallest corner had the point been outside.

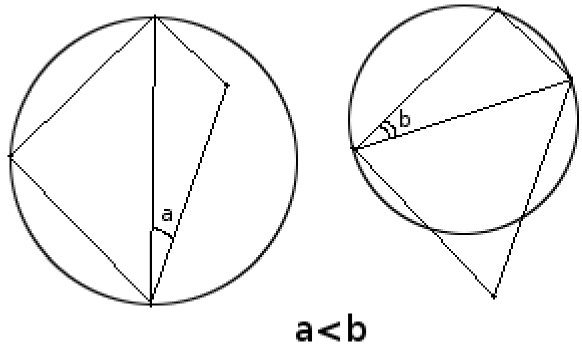

Fig.2 Visual proof that an outside point generates a larger minimal angle.

By generalizing, in order to maximize the minimum angle from all the triangles, all the points must lie outside all circumscribed circles of all the triangles generated. The generalization is valid, because any four points can be validly triangulated just by flipping an edge and maintaining the outside border intact. Another explanation is that by flipping an illegal edge (the illegal edge that generates the smallest angle in the graph) the smallest angle will become greater. With every edge flip the smallest overall angle is increased and the algorithm finishes when no illegal edges remain.

The Voronoi tessellation for a set of points consists of cells around each data point, such as if a random point lies inside one of the cells, it is closer to the cell's point than to any other data point. This means that the Voronoi vertexes (the point where at least three cells meet) are the centers of the circles formed by the three data points (or more if the points are co-circular) that form the neighboring cells. This generates just the condition for Delaunay triangulation, that all points are outside (further away) of the circle formed by any triangle.

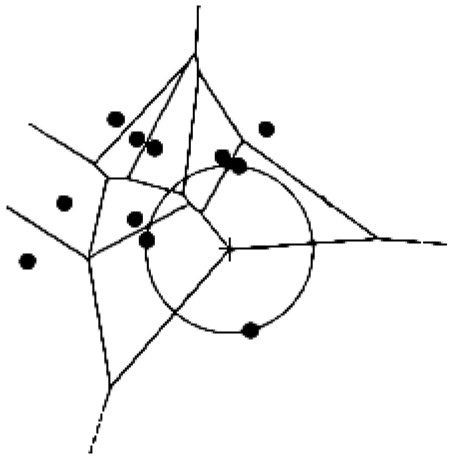

Fig.3 Visual proof that a Voronoi vertex is the center of the circumscribed circle of neighboring data points

The Voronoi graph and Delaunay triangulation are dual: the Delaunay triangulation can be calculated from the Voronoi graph by connecting all neighbors. The effect of this is that any edge will be connected to its closest points (because a Voronoi cell states that all points from the respective region are closest to the respective point).

The conclusion is that if we know the Delaunay triangulation, we already know the best candidate for the outside edges.

Another intuitive proof is that if a different point, other than the one to which the border edge forms a triangle, would be closer to the edge that point would lay inside the Delaunay simplex, either isolated or causing edge intersections.

Knowing this, we generate an algorithm that selects the best candidates for the exterior edges in $O(1)$ time, and removes the exterior edges, appending to the hull the other two edges of the simplex triangle.

The unfiltered algorithm has flaws, as a single point can be best candidate for multiple edges, making the result dependent of the order of traversal. In order to eliminate this inconvenient we can remember the distance from a candidate to the edge and only erode the triangle with the minimal height.

The triangulation needs $O(n \log n)$ time, using a divide-et-impera method [5]. There are at most n points left to add to the border in $O(1)$ time, giving a total complexity of $O(n \log n) + O(n)$.

This way we obtain identical results to the originally proposed algorithm but with better complexity and more rigorous definition.

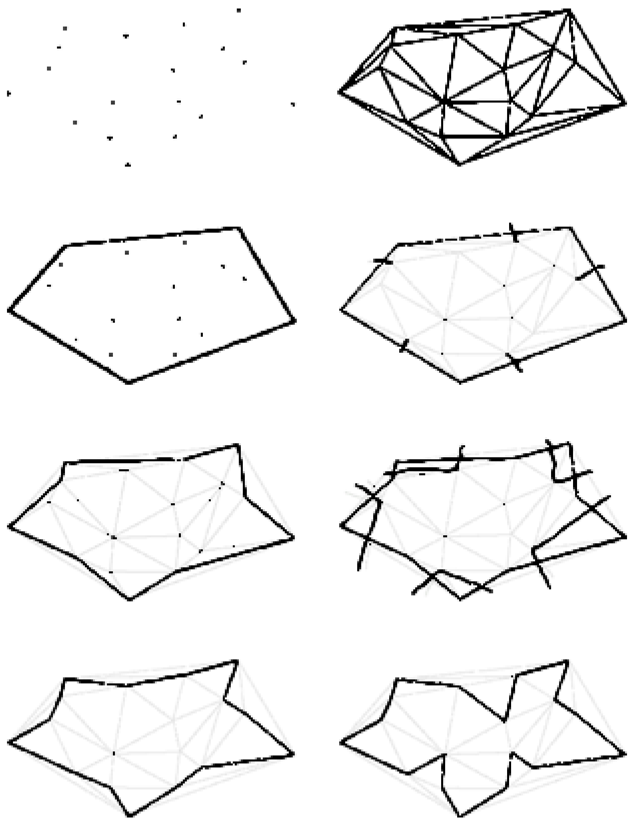

Fig.4 Steps for generating the Beta-Shape using Triangle Erosion.

As the Delaunay triangulation is unique for non-co-circular points, the algorithm offers a single solution. In order to avoid multiple solutions for the case of co-circular data points, we can propose to select the top and left most point in case of a conflict.

6 Natural Flow Beta Shape

However, the algorithm can be refined further, in order to give a result that encourages the erosion of large spaces. Instead of walking through all the border edges, it would intuitively be better to only refine the largest candidate, this means that only one point is added in a single iteration.

This ads an extra ordered list to the complexity for the exterior edges, forming the border, with insertion time of $O(\log n)$ and popping in $O(1)$, thus maintaining the overall complexity.

The results are more natural than those of the original algorithm, following the shape generated by the points (flowing with the geometry).

Situations like the one depicted in Fig. 5 can appear if the document contains white space or separators [7] and the segments are easily interpretable by a classification algorithm (multicolumn, strange layout...).

ISBN: 978-960-474-351-3

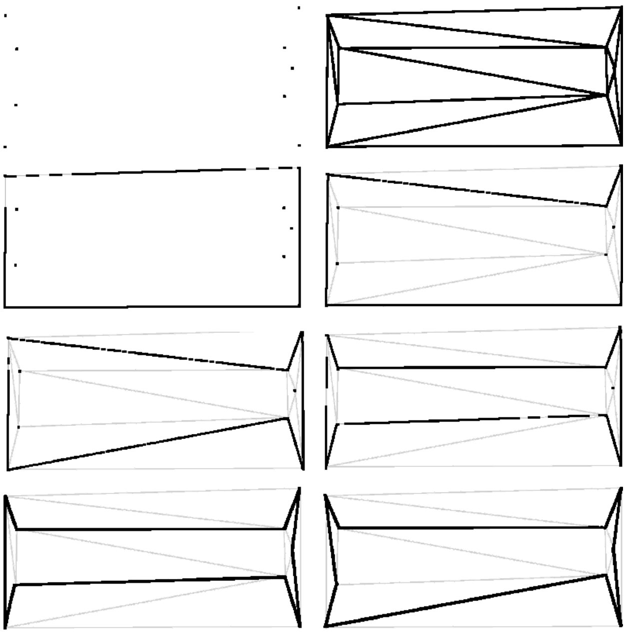

Fig.5 The bottom images are the result of the proposed (left) and old(right) algorithms.

7 Conclusion

We propose an alternative way to compute the beta shape, reducing the overall complexity. Also, a more naturally flowing geometry can be generated with a small improvement.

The method can be generalized for the N-dimensional case, as the Voronoi can be calculated for any dimension, at the expense of complexity (but can be simplified at an initial stage), and an N-dimensional triangulation can result by connecting the neighbors through the Voronoi edges.

The result is a single geometry, at high resolution, useful for containing spread data as close as possible, useful for many of the applications of computational geometry.

References:

[1] C.A. Boiangiu, The Beta-Shape Algorithm for Polygonal Contour Reconstruction, CSCS14, *The 14th International Conference on Control System and Computer Science*", 2003

[2] C.A. Boiangiu, M. Zaharescu, I. Bucur Building Non-Overlapping Polygons for Image Document Layout Analysis Results, *The Proceedings of Journal ISOM*, Vol. 6, No. 2, 2012, pp. 428-436.

[3] F. Cazals, J. Giesen, M. Pauly, A. Zomorodian, Conformal Alpha Shapes, *Eurographics Symposium on Point-Based Graphics*, 2005, pp. 55-61

[4] S. Har-Peled, On the Expected Complexity of Random Convex Hulls, *Duke University, CG*, November 24, 2011

[5] S. Peterson, Computing Constrained Delaunay Triangulations in the Plane, *University of Minnesota Undergraduate*

[6] C.A. Boiangiu, I. Bucur, A.I. Dvornic, The Beta-Star Shape Algorithm for Document Clipping, *Annals of DAAAM for 2008, Proceedings of the 19th International DAAAM Symposium, DAAAM Internationa*l (Vienna, Austria), 2008, pp. 0127–0128

[7] C.A. Boiangiu, D.C. Cananau, A.C. Spataru, Modern Approaches in Detection of Page Separators for Image Clustering. *WSEAS Transactions on Computers*, Vol. 7, No. 7, 2008, pp. 1071 – 1080