International Journal of Computer Science and Engineering (IJCSE)
ISSN(P): 2278-9960; ISSN(E): 2278-9979
Vol. 3, Issue 2, Mar 2014, 9-18
© IASET

International Academy of Science, Engineering and Technology
Connecting Researchers; Nurturing Innovations
IASET

A REVIEW ON DELAUNAY TRIANGULATION WITH APPLICATION ON COMPUTER VISION

SIMENA DINAS & JOSÉ MARÍA BAÑÓN

Department of Computer Engineering, School of Systems, Universidad Del Valle, Valle, Colombia

ABSTRACT

Delaunay Triangulation is formed by a net of triangles that guarantee one property: the circumcircle of each triangle contains only the vertices of the triangle; then, there are not any point of other triangle inside the circum-circle; this property makes the Delaunay triangulation unique. Taking advantage of the uniqueness, Delaunay Triangulation has been used to modeling several problems in different fields; for instance, in this document, has been used to modeling computer vision problems. Rather than solve one problem, this is based on propose to Delaunay Triangulation as an alternative for different application on the area. The preliminary result are the extraction of the features from an image in order to construct the set of points from Delaunay Triangulations, usually called cloud of points from a Delaunay Triangulation. The Delaunay triangulation constructed, shows some similarities between the triangulation and the image; for instance; the result of overlapping the image and the Delaunay Triangulation shows the silhouette from contrasting objects from the image. The question is how can be used this information in order to understanding the image geometry?

KEYWORDS: Cloud of Points, Computer Vision, Delaunay Triangulation, Feature Extraction

I. INTRODUCTION

A Delaunay Triangulation is a triangle net in which every triangle satisfies the Delaunay condition. The Delaunay condition states that the circumcircle of a triangle includes only the vertex of the triangle. In other words, the circumcircle does not contain any vertex of other triangles. Delaunay Triangulation was created in 1934 by Boris Nikolaevich Delone (1890 - 1980); it is widely used in computational geometry. Basically, it is a net of triangles that satisfy a property: for each circumcircle created for a Delaunay triangle, there are not any vertices of other triangles inside [dCvO08]. This property can be extended to edges and tetrahedral in three dimensions. The most common methods to construct a Delaunay Triangulation are: Lawson method [Law 77], Bowyer method [Bow81] and Watson method [Wat81]. In addition, every simplex (triangle tetrahedron edge) satisfying the circumcircle property is usually called a Delaunay simplex; moreover, the union of all simplices produces the convex hull. Every Delaunay Triangulation has a dual graph called a Voronoi Diagram, which is formed by using the circumcenters of each Delaunay triangle.

This document is organized as follows: Delaunay Triangulation is described in section 2. Delaunay Triangualtion and Tetraherdralization are developed in section 3, whereas in Sections 4 is explained the application on Computer vision using Delaunay Triangulation. In section 5 the conclutions are drawn. Finally, acknowledments and references are detailed in the last part of the document.

II. DELAUNAY TRIANGULATION

Delaunay Triangulation is a structure widely used in computational geometry and extended to others multi-purpose areas. Some applications of Delaunay Triangulation include: stereo data [BCFM92]; video compression [VP09]; terrain modeling [KCC99; LR95]; networking [LNS01] and [WLC07]; reconstruction [RELP00]; protein folding [OL08; ZCVT97]; spatial clustering [YC10]; boundary detection [LNS08]; among others.

A Delaunay Triangulation connects the nearest neighbors in a neighborhood; hence it can be used to model the collision detection problem. The Delaunay Triangulation structure is efficient due to each point in the triangulation representing an object in the environment, and then it is checked against neighboring objects to preserving the Delaunay structure and the edges connection. However, the movement of at least one point in a Delaunay Triangulation can produce changes in the structure, and, consequently, it is necessary to update the Delaunay triangulation in order to model dynamic phenomena. Therefore, [GR04; DM08; ZSW+10] are Surveys over updating Delaunay Triangulations.

A. Triangulation

The goal of a triangulation is to produce a mesh. Meshes are a common way to represent continuous surfaces. The input of a Triangulation is a set of points and the result is a set of triangles or linked edges without overlapping [Zim05]. Building a Triangulation has different problems, but the starting point and the dispersion should not be important. Some important applications of triangulation are Digital Terrain Modeling -DMT Generation, Feature Surface Modeling, Computer Graphic, Scientific Visualization, Robotic, Computer Vision, Image Synthesis, Mathematics, and Natural Science. Several proposals of triangulation have been developed [PS85]; [VHS05]; [SAH06]; Greedy Triangulation [DDMW94], Triangulation of Garey [GJPT78], Radial Sweep [HD06]; [MW82] and Delaunay Triangulation [dCvO08].

Greedy Triangulation [DDMW94] is used to triangulate simple polygons by linking together closest points; as a result, shortest edges are generated for the set of points. Greedy Triangulation is not a Delaunay Triangulation, but is an approximation for a Minimum Weight Triangulation Problem.

Radial Sweep has four steps [HD06]; [MW82]: i) to choose the centroid in a set of points *P*, then, linking together the centroid point with every point in *P*. ii) Sort and order the points by orientation and distance (except the centroid), then, link them all together in star-shape without crossing, iii) to add triangles to form a convex shape (convex hull), join together the points inside and outside. A triangulation is created, but it may or may not be a Delaunay Triangulation. iv) Finally, is required to arrange the points focused on their angles in order to produce a correct triangulation. For the set of points *P* shown in figure 1a a Delaunay Triangulation is created in figure 1b. A convex hull for P is depicted in figure 1c. Whereas figure 1d depicts the circumcircles, figure 1e shows the centers for each circumcircles. Finally, the Voronoi Diagrams is represented in figure 1f.

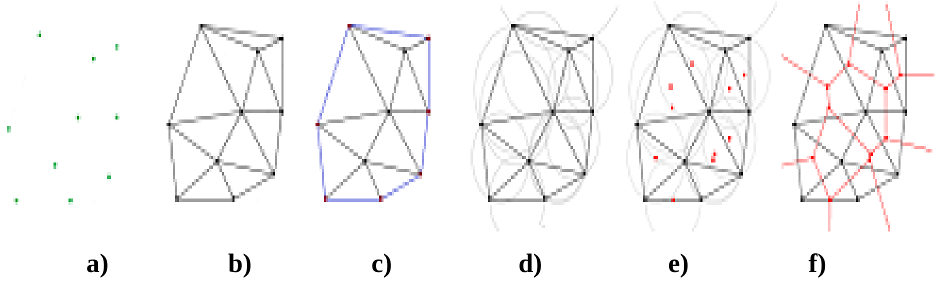


Figure 1: a) Set of Points P. b) Delaunay Triangulation of P. c) Convex Hull of P d) Circumcircles of the Delaunay Triangulation of P. e) Circumcircles and Circumcentres of the Delaunay Triangulation of P f) Voronoi Diagram and Delaunay Triangulation of P

B. Delaunay Triangulation Algorithms

In figure 2a three new triangles are created when a new point p falls inside a triangle, whereas two new triangles are created if p falls into an edge (see figure 2b). Two adjacent triangles are shown in figure 2c four new triangles are created as shown in figure 2d. Because there are not any overlapping triangles in a Delaunay triangulation, new ones replace old ones.

Incremental Algorithms

Incremental Insertion: The algorithm has three parts: i) the initialization, ii) the triangulation and iii) the finalization.

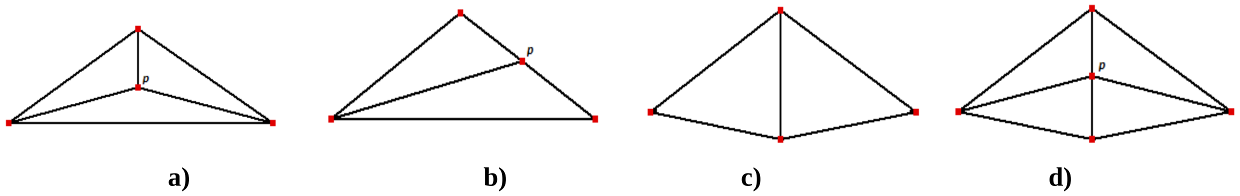


Figure 2: a) Inserting a Point P when it Fall into the Triangle. b) Inserting a Point P when it Fall into an Edge c) Two Triangles Sharing an Edge (Adjacent Edge) d) Inserting a Point P when it Fall into an Adjacent Edge

In the initialization a triangle *t* is constructed that include all points in *P*, but the vertices of *t* are not included in *P*. For the triangulation, the first point is inserted in the triangle *t*, and three new triangles are built (*t*1, *t*2 and *t*3), then *t* is replaced by the new triangles (see figure 2a). Inserting the second point, can be divided into two ways: i) (a) To find the triangle (*t*1, *t*2 and *t*3) in which the point is inside, then three new triangles are built, and the outside triangle (*t*i; 1<= i <= 3) is deleted. (b) If the point is localized in an edge, then is required to find (one or two) triangles in which the edge is involved. If a unique triangle is found, two new triangles will replaced the old one (see figure 2b). If two triangles are found, the pair of triangles will be deleted, and four new triangles will be built (for each triangle, two new ones are constructed) as depicted in figures 2c and 2d. Finally, it is necessary to review the circumcircle condition for each new triangle, and repeat the way to insert the second point [Law77]. ii) To find all triangles whose circumcircle contains the new point, then delete them and the cavity is created, finally join all the vertices in the cavity with the new point [Bow81; Wat81]. In the last part, the triangles that include the vertex of the first triangle is deleted.

Sweepline Algorithms

Plane Sweep Algorithm: A line sweeps the points, when a point is found by the line this is inserted in the triangulation. It only works in two-dimensions.

Incremental Construction Algorithms

Step-by-Step Algorithm: This implementation is based on the uniqueness property of a Delaunay Triangulation. In each step, one new edge or triangle Delaunay is added until completing all points in the set *P*. First, to find a Delaunay edge, then, create a new triangle Delaunay by searching a point *p* [HD06].

Grid Algorithm: To find a point near to the center of the complete set of points, this point is labeled *one*. A first edge is built with the closest point and labeled *two*, the next step is to create the first triangle, following the right-hand rule, it is labeled *three* and the next point is labeled *four* and so. The label is deleted when the point is completely linked or is the center of a star-shape figure [FP93]. An improved algorithm was constructed based on a higher subdivision of the grid [PLK05].

Local Improvement Algorithms

Flipping Algorithm: An arbitrary triangulation is constructed; it is optimized to generate a Delaunay Triangulation.

Divide & Conquer Algorithms

Divide & Conquer: This is a recursive way to calculate Delaunay. The area is subdivided into two parts, and the triangulation is computed recursively in each part. Finally, both triangulations are merged together; however, merging is the most complex step [HD06].

Dewall Algorithm: The name is short for Delaunay Wall. It is a Divide and Conquer algorithm, which computes the simplices between the subdivisions before, in other words, the algorithm merges to calculate recursively the Delaunay Triangulation [CMS97].

Properties of Delaunay Triangulation

Local Empty-Circle: A circumcircle is a unique circle passing through each vertex of the triangle in a Delaunay Triangulation *DT (P)*. The circle contains no other point from the set of points *P*, (see figures 3a and 3b).

Max-Min Angle: It was introduced by Sibson in [Sib78]. For each quadrilateral in Delaunay Triangulation *DT (P)*, two possible triangulations can be produced, the correct ones maximize the minimum of the six internal angles (see figures 3c and 3d).

Uniqueness: Delaunay Triangulation *DT(P)*, for a set of points *P*, is unique, except for a set of four co-circular points; it is called a degenerate case, for instance: a square (in two dimensions) or a cube (in three dimensions), where there are different options to make the division as shown in figures 3e and 3f. A degenerate case happens when four or more points fall into a circumcircle (in two dimensions) or five or more points fall into a circumsphere (in three dimensions). For more details of degenerated cases, see [Ver10].

Boundary: The external edges of a Delaunay Triangulation *DT (P)*, make the Convex-hull for *P*. The Local Empty Property is depicted in figure 3a, it is due to circumcircles enclosing more than three points, whereas in figure 3b, for every triangle, the circumcircle encloses three points.

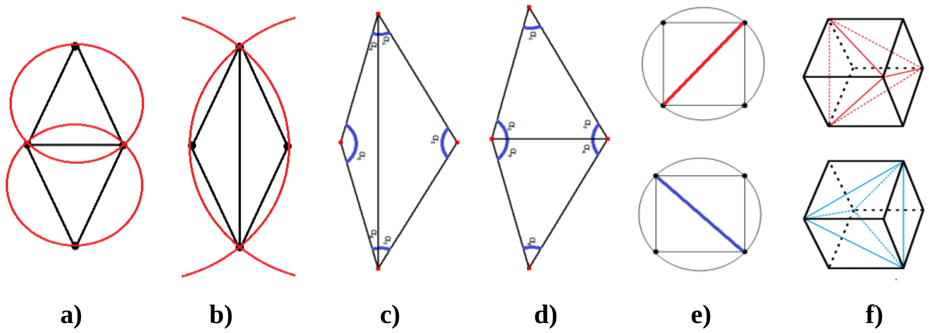
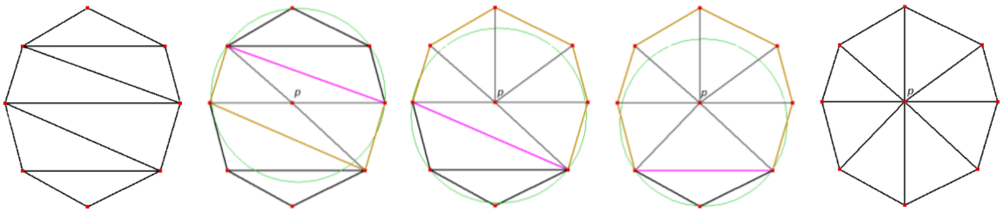


Figure 3: a) Delaunay Triangles b) Non-Delaunay Triangles c) Bad Angles in a Delaunay Triangulation. d) Good Angles in a Delaunay Triangulation. e) No-Unique Delaunay Triangulation f) No-Unique Delaunay Tetrahedralization

IV. DELAUNAY TRIANGULATION AND DELAUNAY TETRAHEDRALIZATION

To let P^2 be a set of points in R^2 then to let T^2 be a Triangulation of P^2, T^2 is a Delaunay Triangulation of P^2, DT^2 (P^2) if and only if for every triangle of T^2, the circumcircle does not contain any other points of P^2 (see figures 4a, 4b, 4c, 4d and 4e). Delaunay Tetrahedralization is based on extending Delaunay Triangulation to three dimensions. To let P^3 be a set of points in R^3 then to let T^3 be a Tetrahedralization of P^3, T^3 is a Delaunay Tetrahedralization of P^3, DT^3 (P^3) if and only if for every tetrahedron of T^3, the circumsphere contains no other points of P^3.

Sequence of insertion of a point p in a Delaunay Triangulation.



a) b) c) d) e)

Figure 4: a) Initial Configuration b) Inserting a Point P c) Flipping Edges d) Last Flipping of an Edge e) Final Configuration

In figure 4a: the initial Delaunay Triangulation of the set of eight points. In fgure 4b: a new point *p* is inserted in the triangulation. Because *p* falls into an edge, two adjacent triangles are affected; accordingly, four new triangles are created (two for each affected triangle). New edges are evaluated, the flipping process requires evaluation by using in Circle function, and the circumcircle is shown. Figure 4c shows the evaluation of a triangle; whereas figure 4d depicts the evaluation of other triangle. The circumcircle of the flipping edges is depicted by colors. The final configuration of a Delaunay Triangulation is shown in figure 4e. Some steps have been ignored.

Delaunay Triangulation and Tetrahedralization Tests

$$orientation2D(p_0,p_1,p_2) = \begin{vmatrix} x_0 & y_0 & 1 \\ x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \end{vmatrix} = \begin{vmatrix} (x_0-x_2) & (y_0-y_2) \\ (x_1-x_2) & (y_1-y_2) \end{vmatrix} \tag{1}$$

$$inCircle(p_0,p_1,p_2,p_3) = \begin{vmatrix} x_0 & y_0 & {x_0}^2+{y_0}^2 & 1 \\ x_1 & y_1 & {x_1}^2+{y_1}^2 & 1 \\ x_2 & y_2 & {x_2}^2+{y_2}^2 & 1 \\ x_3 & y_3 & {x_3}^2+{y_3}^2 & 1 \end{vmatrix} = \begin{vmatrix} (x_0-x_3) & (y_0-y_3) & (x_0-x_3)^2+(y_0-y_3)^2 \\ (x_1-x_3) & (y_1-y_3) & (x_1-x_3)^2+(y_1-y_3)^2 \\ (x_2-x_3) & (y_2-y_3) & (x_2-x_3)^2+(y_2-y_3)^2 \end{vmatrix} \tag{2}$$

$$orientation3D(p_0,p_1,p_2,p_3) = \begin{vmatrix} x_0 & y_0 & z_0 & 1 \\ x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ x_3 & y_3 & z_3 & 1 \end{vmatrix} = \begin{vmatrix} (x_0-x_3) & (y_0-y_3) & (z_0-z_3) \\ (x_1-x_3) & (y_1-y_3) & (z_1-z_3) \\ (x_2-x_3) & (y_2-y_3) & (z_2-z_3) \end{vmatrix} \tag{3}$$

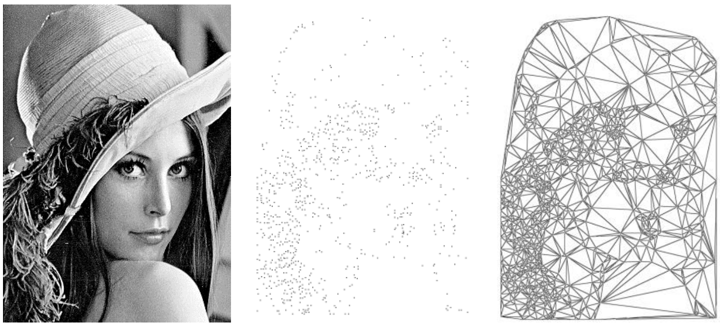
$$\begin{aligned} inSphere(p_0,p_1,p_2,p_3,p_4) &= \begin{vmatrix} x_0 & y_0 & z_0 & {x_0}^2+{y_0}^2+{z_0}^2 & 1 \\ x_1 & y_1 & z_1 & {x_1}^2+{y_1}^2+{z_1}^2 & 1 \\ x_2 & y_2 & z_2 & {x_2}^2+{y_2}^2+{z_2}^2 & 1 \\ x_3 & y_3 & z_3 & {x_3}^2+{y_3}^2+{z_3}^2 & 1 \\ x_4 & y_4 & z_4 & {x_4}^2+{y_4}^2+{z_4}^2 & 1 \end{vmatrix} \\ &= \begin{vmatrix} (x_0(t)-x_4(t)) & (y_0(t)-y_4(t)) & (z_0(t)-z_4(t)) & l_{0-4}(t) \\ (x_1(t)-x_4(t)) & (y_1(t)-y_4(t)) & (z_1(t)-z_4(t)) & l_{1-4}(t) \\ (x_2(t)-x_4(t)) & (y_2(t)-y_4(t)) & (z_2(t)-z_4(t)) & l_{2-4}(t) \\ (x_3(t)-x_4(t)) & (y_3(t)-y_4(t)) & (z_3(t)-z_4(t)) & l_{3-4}(t) \end{vmatrix} \end{aligned} \tag{4}$$

Delaunay Triangulation orientation and inCircle tests: To calculate a Delaunay Triangulation, a circumcircle test is required; however, the triangle has to be in counterclockwise orientation. The orientation can be calculated using equation 1, whereas the inCircle test can be calculated by using equation 2.

Delaunay Tetrahedralization orientation test: Analogously, in three dimensions, the orientation test can be calculated using the determinant (see equation 3) and the circumsphere test, usually called inSphere test, which is used to calculate a valid tetrahedron, it is shown in equation 4.

V. AN APPLICATION OF DELAUNAY TRIANGULATION ON COMPUTER VISION

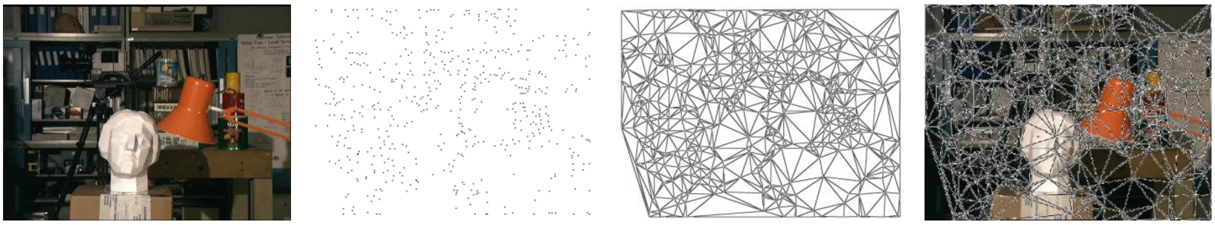
In general terms, the Delaunay Triangulation can be proved with a cloud of points; however, data can be hardly understandable and traceable. A Delaunay Triangulation was developed by using features extracted from the figure 5a in order to depict an easily identifiable triangulation. The features extracted were corners or interesting points from the image.



a) b) c)

Figure 5: a) The Lena Picture. b) Corner Extraction of Lena Picture c) Delaunay Triangulation by Using Cloud of Points from Lena Picture

Preliminary results on Delaunay Triangulation include implementation and tests. In order to show the Delaunay triangulation; it gets salient features from an image to triangulate them. In figure 5a the picture of Lena is depicted. Figure 5b shows the set of points extracted from the picture of Lena. Finally, using the set of points, a Delaunay Triangulation is constructed (see figure 5c).



a) b) c) d)

Figure 6: a) This Figure Shows an Image Called Tsukuba b) This Figure Represents the Tsukuba Cloud of Points Extracted from the Image c) This Figure Represents the Tsukuba Triangulation by Means of the Cloud of Points Extracted from the Image d) The Tsukuba Picture Superimposes with the Triangulation of the Figure 6c

Delaunay Triangulation for Tsukuba image (see figure 6a) is depicted in figure 6c; it is created by using the cloud of points extracted from the image (see figure 6b). The combination of the Tsukuba image and the Delaunay Triangulation from the image is shown in 6d; there are big triangles representing the human form (white) and the lamp (orange) because the colors are homogeneous and there are fewer features to extracting. Similarly, there are big triangles below the lamp because there are fewer features. In contrast, the rest of the image has small triangles because more features can be extracted. Moreover, the figure 6d shows the silhouette from the contrasting objects: the human form and the lamp; it is because interesting points were found in the border of the object, maybe due to the intensity changes; however, it is casual that the silhouette form being clearly identifiable.

Two interesting problems in computer vision are the Correspondence Estimation and the Motion Estimation. Correspondence estimation is based on establishing a pair of corresponding points from to slightly different images, which represent the same point in the real world. The pair of images are taken from different points of view from correspondence estimation; in contrast, the definition of motion estimation is contrary, the images are taken from the same point of view and the difference between them is the time of the capture; usually, the images from motion estimation are called video or images sequence. The questions are how can features extracted from images and Delaunay triangulation be used to track the objects inside the image? How can the Delaunay triangulation from stereo images be used to know either the depth and improve the disparity map construction or the objects motion by using motion vectors? This paper is not trying to solve these questions, but is trying to identify the relationship between the Delaunay Triangulation and stereo images. Several authors have explored the Delaunay Triangulation on 3D Reconstruction [DGH01], [LYG+09]; however, the work on stereo correspondence in order to improve the disparity and depth estimations is poor [CST+08].

VI. CONCLUSIONS

Delaunay Triangulation has been used in several areas because its versatility to adapt different nature structures. Since the connection between similar objects is important for different problems in different areas, the capacity of Delaunay triangulation for maintains together the closest objects is significant. Even though the most known application of Delaunay Triangulation is the mesh, this geometry have been used in dozens of applications including medicine, terrain modeling, stereo correspondence, motion, video compression, amongst others.

Correspondence estimation involves the geometry from images; in the same way, Delaunay triangulation is widely known in geometry and widely used in computational geometry. Both can be combined to explore the advantages from stereo images in order to improve the disparity map construction. On the other hand, motion estimation requires geometrical information from images as well; for instance, can be supposed that small perturbations can produce slightly changes in the triangulation, but there are not guarantee about this because the problem is not the movement of the objects, there are more aspects involved in the movement as direction and the position of the object after and before the movement with respect to the rest of objects. This approximation is an attempt to demonstrate the potential work that can be developed using one of the most complex and widely used geometric structures. Delaunay is not a panacea, but the results in other areas combined with reliability and uniqueness of the structure can help improve performance in computer vision and other areas.

ACKNOWLEDGEMENTS

The authors would like to thank to COLCIENCIAS and Universidad del Valle for the help.

REFERENCES

1. [BCFM92] E. Bruzzone, M. Cazzanti, L. De Floriani, and F. Mangili. Applying two-dimensional delaunay triangulation to stereo data interpolation. In Proceedings of the Second European Conference on Computer Vision, ECCV ’92, pages 368–372, London, UK, 1992. Springer-Verlag.

2. [Bow81] A. Bowyer. Computing dirichlet tessellations. The Computer Journal, 24(2):162–166, February 1981.
3. [CMS97] P. Cignoni, C. Montani, and R. Scopigno. Dewall: A fast divide & conquer Delaunay triangulation algorithm in ed, 1997.
4. [CST+08] Chao-I Chen, Dusty Sargent, Chang-Ming Tsai, Yuan-Fang Wang, and Dan Koppel. Stabilizing stereo correspondence computation using Delaunay triangulation and planar homography. In Proceedings of the 4th International Symposium on Advances in Visual Computing, ISVC '08, pages 836–845, Berlin, Heidelberg, 2008. Springer-Verlag.
5. [dCvO08] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. Computational Geometry: Algorithms and Applications. Springer-Verlag, 3rd edition, 2008.
6. [DDMW94] M. T. Dickerson, R. L. S. Drysdale, S. A. McElfresh, and E.Welzl. Fast greedy triangulation algorithms. In Proceedings of the tenth annual symposium on Computational geometry, SCG'94, pages 211–220, New York, NY, USA, 1994. ACM.
7. [DGH01] Tamal K. Dey, Joachim Giesen, and James Hudson. Delaunay based shape reconstruction from large data. In Proceedings ofthe IEEE 2001 symposium on parallel and large-data visualizationand graphics, PVG '01, pages 19–27, Piscataway, NJ, USA, 2001. IEEE Press.
8. [DM08] O. Devillers and P. Machado Manh˜aes de Castro. State of the art: Updating delaunay triangulations for moving points. Rapport de recherche RR-6665, INRIA, 2008.
9. [FP93] T.-P. Fang and L. A. Piegl. Delaunay triangulation using a uniform grid. Computer Graphics and Applications, IEEE, 13(3):36–47, May 1993.
10. [GJPT78] M. R. Garey, D. S. Johnson, F. P. Preparata, and R. E. Tarjan. Triangulating a simple polygon. Information Processing Letters, 7:175–180, 1978.
11. [GR04] L. J. Guibas and D. Russel. An empirical comparison of techniques for updating Delaunay triangulations. In Proceedings of the twentieth annual symposium on Computational geometry, SCG'04, pages 170–179, New York, NY, USA, 2004. ACM.
12. [HD06] ØHjelle and M. Dæhlen. Triangulations and Applications (Mathematics and Visualization). Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
13. [KCC99] S.-S. Kim, M.-G. Cho, and H.-G. Cho. A geometric compression algorithm for massive terrain data using Delaunay triangulation. WSCG99 Proc, pages 124–131, 1999.
14. [Law77] Ch. L. Lawson. Software for c1 surface interpolation. In Mathematical Software III, J. Rice ed., pages 161–194. Academic Press, New York, 1977.
15. [LNS01] J. Liebeherr, M. Nahas, and W. Si. Application-layer multicasting with delaunay triangulation overlays. Technical report, Charlottesville, VA, USA, 2001.
16. [LNS08] D. Liu, G. V. Nosovskiy, and O. Sourina. Effective clustering and boundary detection algorithm based on Delaunay triangulation. Pattern Recogn. Lett., 29:1261–1273, July 2008.

17. [LR95] R. Lattuada and J. Raper. Applications of 3d Delaunay triangulation algorithms in geoscientific modelling. In Proc. 3rd Nat. Conf. GIS Research UK, Apr 1995.

18. [LYG+09] Sizhe Lv, Xubo Yang, Lixu Gu, Xiaojin Xing, Lei Pan, and Min Fang. Delaunay mesh reconstruction from 3d medical images based on centroidal voronoi tessellations. Computational Intelligence and Software Engineering 2009 CiSE 2009 International Conference on, pages 1–4, 2009.

19. [MW82] A. Mirante and N. Weingarten. The radial sweep algorithm for constructing triangulated irregular networks. IEEE Comput. Graph. Appl., 2:11–21, March 1982.

20. [OL08] Z. Ouyang and J. Liang. Predicting protein folding rates from geometric contact and amino acid sequence. Protein Science, 17(7):1256–1263, 2008.

21. [PLK05] S. H. Park, S. S. Lee, and J. H. Kim. The Delaunay triangulation by grid subdivision. In Osvaldo Gervasi, Marina L. Gavrilova, Vipin Kumar, Antonio Lagan`a, Heow Pueh Lee, Youngsong Mun, David Taniar, and Chih Jeng Kenneth Tan, editors, Computational Science and Its Applications – ICCSA 2005, volume 3482 of Lecture Notes in Computer Science, pages 1033–1042. Springer Berlin / Heidelberg, 2005.

22. [PS85] F. P. Preparata and M. I. Shamos. Computational Geometry: An Introduction. Springer-Verlag New York, Inc., New York, NY, USA, 1985.

23. [RELP00] A. Rodr´ıguez, J. Espadero, D. L´opez, and L. Pastor. Delaunay surface reconstruction from scattered points. In Gunilla Borgefors, Ingela Nystr¨om, and Gabriella di Baja, editors, Discrete Geometry for Computer Imagery, volume 1953 of Lecture Notes in Computer Science, pages 272–283. Springer Berlin / Heidelberg, 2000. 10.1007/3-540-44438-6 23.

24. [SAH06] D. Shojaee, A. A. Alesheikh, and H. Helali. Triangulation for surface modelling, 2006.

25. [Sib78] R. Sibson. Locally equiangular triangulations. The Computer Journal, 21(3):243–245, 1978.

26. [Ver10] E. Verbree. Delaunay tetrahedralizations: Honor degenerated cases. In The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, pages 69–72, 2010.

27. [VHS05] M. Varshosaz, H. Helali, and D. Shojaee. The methods of triangulation, 2005.

28. [VP09] T. Vom´aˇcka and P. Puncman. A novel video compression scheme based on kinetic Delaunay triangulation. In Algoritmy 2009: 18th Conference on Scientific Computing, pages 372–381, Bratislava, 2009. Slovak University of Technology.

29. [Wat81] D. F. Watson. Computing the n-dimensional Delaunay tessellation with application to voronoi polytopes. The Computer Journal, 24(2):167–172, February 1981.

30. [WLC07] Ch.-H. Wu, K.-Ch. Lee, and Y.-Ch. Chung. A Delaunay triangulation based method for wireless sensor network deployment. Comput. Commun., 30:2744–2752, October 2007.

31. [YC10] X. Yang and W. Cui. A novel spatial clustering algorithm based on Delaunay triangulation. Journal of Software Engineering and Applications, JSEA, 3(2):141–149, 2010.

32. [ZCVT97] W. Zheng, S. J. Cho, I. I. Vaisman, and A. Tropsha. A new approach to protein fold recognition based on Delaunay tessellation of protein structure. In Pacific Symposium on Biocomputing, 1997.

33. [Zim05] H. Zimmer. Voronoi and Delaunay techniques. Lecture Notes in Computer Science VIII, 2005.

34. [ZSW+10] Y. Zhou, F. Sun, W. Wang, J. Wang, and C. Zhang. Fast updating of Delaunay triangulation of moving points by bi-cell filtering. Computer Graphics Forum, 29(7):2233–2242, 2010.