Indian Journal of Science and Technology, *Vol 7(4), 426–429, April 2014*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Delaunay Edge Detection using Modified Star Formation in Two Dimensional Data

R. Mukunthan[1*] and N. Sairam[2]

[1]School of Computing, SASTRA University, Thanjavur, TamilNadu, India; mukunthan2010@yahoo.com
[2]School of Computing, SASTRA University, Thanjavur, TamilNadu, India; sairam@cse.sastra.edu

Abstract

A new method for detecting Delaunay edge by modifying the links in the star of a vertex is proposed. This is based on selecting vertex points of the input triangulation in such a way that the star formed from the selected point should belong to the given input set *S*. That star should not have any convex hull point and the edges connecting the selected vertex. The edges formed in the proposed method based on star formation are Delaunay edges since it satisfies the empty circle property. This is experimentally verified using two dimensional input data. Finally, Delaunay triangulation is obtained by joining the remaining edges which are validated and verified using the circumcircle property of Delaunay triangulation.

Keywords: Convex Hull, Delaunay Triangulation, Star Formation

1. Introduction

The Delaunay edge detection method determines whether the triangulation of the input set is in Delaunay or not. The Delaunay triangulation, its dual Voronoi Diagram and Convex Hull are essential structures for the field of Computational Geometry. Particularly Delaunay triangulation has vast usage. A triangulation in the input set *S* is said to be Delaunay if the triangle P_t formed by the three points S_1, S_2 and S_3 such that $P_t = \{S_1, S_2, S_3\} \in S$ and the triangle P_t should satisfy empty circle (circumcircle) property. If set of all triangles formed in an input triangulation satisfies this property, then the input triangulation is called Delaunay triangulation DT(*S*) of the input set *S*. DT(*S*) is used for the approximation of a terrain model[1]. It is also used to construct convex hull *S*. of Zhou[2] extended Delaunay triangulation for wireless sensor network.

The proposed method checks the input triangulation by selecting $S_p \,|S_p \in S;\, p = 1, 2 \ldots|\, S_1$, obtaining stars and links for S_p and verifying empty circle property between the selected S_p and all its points on the link. The edge connecting S_p and a point in the link which satisfies the empty circle property is a Delaunay edge connecting S_p and that point in the link.

2. Related Work

Shewchuk[4] proposed star splaying algorithm for repairing convex hull and Delaunay triangulation. This star splaying algorithm takes an input triangulation and outputs Delaunay triangulation. For mesh generation, Lee[5] used a modified Delaunay triangulation. Lawson's edge flipping algorithm[6] results in a good Delaunay detection. Bowyer[7] proposed efficient algorithm for Delaunay triangulation in higher dimensional space. Renka[8] constructed Delaunay triangulation on the sphere surface using incremental algorithm. It facilitates updating the triangulation by considering dynamic insertion and deletion of points. Sewel[9] used Delaunay triangulation for optimizing mesh generation which is used for Transmission–line Modelling (TLM). This TLM is used for electromagnetic simulation. Hwang[10] proposed a spatial query processing method which is based on elastic transformation using Delaunay triangulation. This is applicable for database system in GIS. The self improving algorithm was proposed by Clarkson[11] to construct the Delaunay triangulation for the given input point set *S*. Dwyer[12] proposed a modification of divide and conquer algorithm for planar Delaunay triangulation. Divide and conquer algorithm

**Author for correspondence*

for Delaunay triangulation was proposed by Guibas and Stolfi. Dwyer's modification results in average case running time of O (n loglog n) for Delaunay construction. Karasick[13] proposed adaptive-precision algorithms and integrate with Guibas-Stolfi algorithm for constructing Delaunay triangulation.

3. Proposed Delaunay Edge Detection Approach

3.1 Overview

The proposed approach provides a modification of star formation. It follows a different approach of star formation when compared with Shewchuk's star splaying method[4]. First, a point $S_p = (x_p, y_p)$ is selected in the input triangulation. Then Euclidean distance between S_p and extreme point of Y co-ordinate is calculated. This distance act as a radius of a circle whose center is S_p. The points (vertices) of the input triangulation which are inside the radius are considered for computing link of the star for S_p. Then empty circle property for edge connecting S_p and a vertex in link is verified. An edge connecting vertex of link and S_p is removed if it doesn't satisfy empty circle property. Only the edges which satisfy empty circle property forms Delaunay edge.

3.2 Algorithm for Delaunay Edge Detection

Algorithm 1 detects Delaunay edges for a given point S_p.

Algorithm 1: Compute Delaunay edge detection

Input: S_p, a point (vertex) whose edges connecting other points in input triangulation which need to be checked for Delaunay property.

1: Let $y_{min} = (x_1, y_1)$ be extreme point having minimum Y co-ordinate.
2: Let $y_{min} = (x_1, y_1)$ be extreme point having maximum Y co-ordinate.
3: Set $S_p = (x_p, y_p)$ as center of the circle.
4: Set $r = \max(|y_{min} - S_p|, |y_{max} - S_p|)/2$ and initialize S';
5: For each point $S_i \mid S_i \in S$; i = 1, 2 ... $|S|$
      If $|S_p - S_i| \le r$ then
        Append S_i in S';
      End if
   End for
6: Set $S_{Star} = \text{getStar}(S_p, S')$

Output: S_{Star} has points which are Delaunay edges intersecting at the vertex S_p.

In Algorithm 1, y_{min} and y_{max} are the points having minimum and maximum Y co-ordinates. Using these points, radius of the circle is determined. The radius of the circle, r is computed in such a way that all neighbouring points for S_p are covered irrespective of point distribution (Step 4). S' containing all points which are inside the radius of coverage with S_p as center. The function getStar is used to find links of the star for S_p. The points in S_{Star} forms star shape when its edges are connected with S_p.

Algorithm 2: Detect edges that satisfies empty circle property

Procedure getStar(S_p, S')
1: Initialize S_t;
2: For each point $S'_i \mid S'_i \in S'$; i=1, 2 ... $|S'|$
    If $|S'_i - S''_j| \ge |S_p - S''_j|$
      Append S'_i in S_t;
    End if
  End for
3: For each point $S_k | k = 1, 2 ....|S_t|$ in S_t do

Set $S_m = \left(\left(\frac{x_p + x_k}{2}\right), \left(\frac{y_p + y_k}{2}\right)\right)$ as midpoint and

$r = |S_p - S_m|$;
If k = 1
    If $(|S_m - S_k| \le r)$ or $(|S_m - S_{k+1}| \le r)$
        $S_t = S_t \setminus S_k$;
      End if
    Elseif $k = |S_t|$
      If $(|S_m - S_{k-1}| \le r)$ or $(|S_m - S_1| \le r)$
        $S_t = S_t \setminus S_k$;
      End if
    Else
  If $(|S_m - S_{k-1}| \le r)$ or $(|S_m - S_{k+1}| \le r)$
        $S_t = S_t \setminus S_k$;
      End if
    End if
End procedure

Algorithm 2 describes the getStar function. The function (procedure) getStar takes S_p and S' from Algorithm 1 as input and returns S_t as output. S_t has all Delaunay edges for S_p. It is noted that this S_t in Algorithm 2 is collected as S_{Star} in Algorithm 1. In step 2 of Algorithm 2, S''_j represents
$S''_j \in S' | S''_j = ((x_p + x_{(j)})/2), ((y_p + y_{(j)})/2); \forall j$,
$j = 1,2 \ldots |S'|; j \neq i$. In step 3, S_k denotes the set of points in S_t.

4. Experimental Result

The proposed algorithm was implemented in Matlab 7.11.0(R2010b). Figure 1 shows the input triangulation with S_p. Figure 2 and Figure 3 shows the processing of proposed algorithm for S_p. Figure 3 shows the result of verifying Delaunay edge (algorithm 1) for neighbouring vertices of Delaunay edges of S_p (from Figure 2). Figure 4 describes the final Delaunay triangulation obtained from input triangulation as a result of the proposed Delaunay edge detection.

5. Conclusion

This work proposed a new way of detecting Delaunay edge and verifying the given triangulation for two dimensional data. This method works independent of Voronoi construction. In future, computing Delaunay edge detection for three dimensional data will be considered.

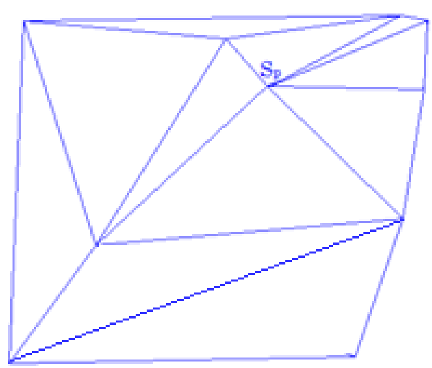

Figure 1. Input triangulation.

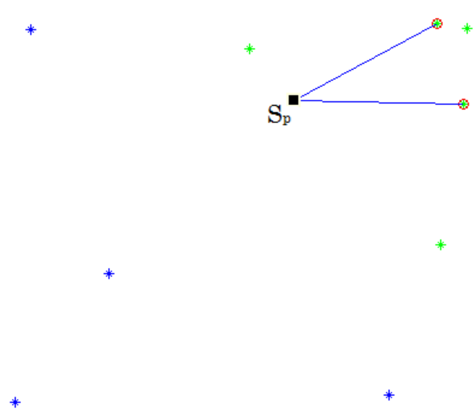

Figure 2. Detecting proper Delaunay edge.

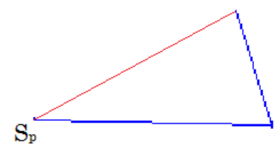

Figure 3. Corrected Delaunay edge shown in red color.

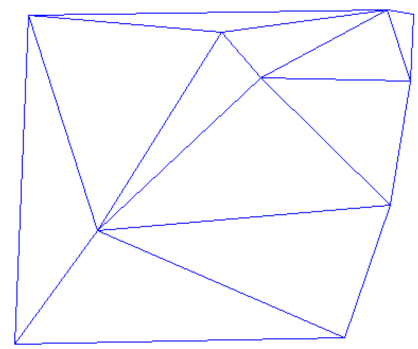

Figure 4. Final Delaunay triangulation.

6. References

1. de Berg M, Cheong O, van Kreveld M, Overmars M. Computational geometry algorithms and applications. 3rd rev. ed. Berlin Heidelberg: Springer-Verlag; 2008.
2. Zhou H, Jin M, Wu H. A distributed delaunay triangulation algorithm based on centroidal voronoi tessellation for wireless sensor networks. Proceedings of the fourteenth ACM international symposium on Mobile ad hoc networking and computing (MobiHoc '13); 2013 ACM, New York. p. 59–68.
3. Chew L P. Constrained Delaunay Triangulation. Algorithmica. 1989; 4:97–108.
4. Shewchuk JR. Star splaying: An algorithm for repairing Delaunay triangulations and convex hulls. Proceedings of the 21st Symposium on Computational Geometry (SoCG'05). 2005; ACM, New York. p. 237–46.
5. Bowyer A. Computing dirichlet tessellations. Comput J. 1981; 24(2):162–66.
6. Lee JF, Romanus DE. Automatic mesh generation using a modified Delaunay Tessellation. IEEE Antenn Propag Mag. 1997 Feb; 39(1):34–35.
7. Lawson CL. Transforming triangulations. Discrete Math. 1972; 3(4):365–72.
8. Renka RJ. STRIPACK: Delaunay triangulation and Voronoi diagram on the Surface of a Sphere. ACM Trans Math Software. 1997; 23(3):416–34.
9. Sewell P, Benson TM, Vukovic A, Cole S. Mesh optimisation methods for unstructured transmission-line modelling. IET Science, Measurement & Technology. 2013; 7(1):32–40.

10. Hwang J-R, Oh J-H, Li K-J. Query transformation method by delaunay triangulation for multi-source distributed spatial database systems. Proceedings of the 9th ACM international symposium on Advances in geographic information systems (GIS '01); 2001. ACM, New York. p. 41–6.
11. Clarkson KL, Seshadhri C. Self-improving algorithms for delaunay triangulations. SCG'08. Proceedings of the twenty-fourth annual symposium on Computational (SCG'08); 2008. ACM, New York. p. 148–55.
12. Dwyer RA. A simple divide-and-conquer algorithm for computing Delaunay triangulations in O(n log log n) expected time. Proceedings of the second annual symposium on Computational geometry (SCG'86); 1986. ACM, New York. p. 276–84.
13. Karasick M, Lieber D, Nackman LR. Efficient Delaunay triangulation using rational arithmetic. ACM Trans Graph. 1991; 10(1):71–91.
14. Guibas L, Stolfi J. Primitives for the manipulation of general subdivisions and the computation of Voronoi. ACM Trans Graph. 1985; 4(2):74–123.