Bulletin of the *Transilvania* University of Braşov
Series I: Engineering Sciences • Vol. 4 (53) No. 1 - 2011

ON THE APPLICATION OF VORONOI DIAGRAMS AND DELAUNAY TRIANGULATION TO 3D RECONSTRUCTION

T.T. COCIAŞ[1] **G. MĂCESANU**[1]
F. MOLDOVEANU[1]

Abstract: *Voronoi diagrams and Delaunay triangulation have many properties that are highly desirable for 3D modelling applications and spatial analysis. Therefore they are considered to be the fundamental in 3D space reconstruction. We have highlighted in this paper a short and fast algorithm to optimally compute the Delaunay triangulation, used in the reconstruction of 3D geometric figures where the complexity of the problem is greater than the classical 2D plane case.*

Key words: *3D Reconstruction, Voronoi diagrams, Delaunay triangulation.*

1. Introduction

In the last years, the 3D reconstruction and modelling of an object has become a topic of interest for several fields of research. Particular attention has been paid on the reconstruction of a realistic scene, which can be applied to a wide range of applications such as robotics, virtual reality, medicine, surveillance and industry, in order to gain a much better analysis of the environment. A tri-dimensional model can be created from a set of topographic cross section [14]. Thus, the reconstruction approach can be roughly classified into two groups: *volume reconstruction* and *surface reconstruction*. The first group can be tackled with the classical solution presented in [8], [16], while the last group can be treated with today's dominant *voxel* technique as in [1], [3]. On the other hand, these techniques can be divided in two completely different approaches: *approximation* and *interpolation*. The approximation approach estimates a surface that passes nearly the original shape presented by Hoppe in [15], while the second approach uses Voronoi diagrams and Delaunay triangulation to find a topological connection between sampled points.

Voronoi diagrams and their duals Delaunay triangulation approach are fundamental for modelling a space from a set of calculated points. They provide a decomposition of the space surrounding the set of points into well shaped cells which can be used to extract proximity information and detect collisions [12], [13].

A drawback is that the methods are time consuming. There were several attempts to optimise the algorithms in order to make

[1] Dept. of Automatics, *Transilvania* University of Braşov.

optimise the algorithms in order to make them faster [5, 6, 11]. The most relevant modification of the Delaunay algorithm, presented in [2], allows the computation of millions of points, making it suitable for many 3D applications. The points needed to reconstruct an object can be obtained either from a disparity image [9], computed from pairs of stereo-images, either from an image range sensor or a laser sensor. The last two methods are time consuming but offer a much better representation of the object of interest.

This paper is organised as fallows. In Section 2 we review the Voronoi diagrams and Delaunay triangulations outlining their essential characteristics. In Section 3 the Delaunay triangulation algorithm is described. Section 4 presents some applications of the concepts to surfaces in the 2D and 3D spaces, respectively. We conclude with a discussion on the performances of the methods.

2. Solving the reconstruction problem

Starting from a set of sampled points, the reconstruction problem can be defined as a method to establish neighbourhood connections between the samples. This geometric construction can be made using Voronoi diagram and its dual Delaunay triangulation.

2.1 Voronoi diagrams. Definition

The Voronoi diagram is easy to describe and, via a duality relationship, it facilitates the description of the Delaunay triangulation. Given a set $\mathcal{P}$ of n points in $\Re^d$, the Voronoi diagram partitions $\Re^d$ into n cells: one cell is associated with each point in $\mathcal{P}$. For point $p \in \mathcal{P}$, we denote the associated Voronoi cell by $V(\mathcal{P})$. The extent of $V(p)$ is simply the entire region of $\Re^d$ whose distance to $\mathcal{P}$ is realized by the distance to p. That is, the set of points that is at least as close to p as it is to any other point $q \in \mathcal{P}$.

The set of Voronoi cells forms a covering of S called the Voronoi diagram of $\mathcal{P}$. The Voronoi diagram gives a very natural definition of the neighbours of a point $p \in \mathcal{P}$.

The Voronoi cells are convex polygons in $\Re^2$, and in higher dimensions they are convex polytopes [10]. Indeed, $V(p)$ can be constructed as the intersection of the $n-1$ half spaces each of which contains point p and is bounded by the orthogonal bisector of $[p, q]$ for some $q \in \mathcal{P}$. The intersection of $d+1$ or more Voronoi cells is either empty or a single point, called a *Voronoi vertex*. A Voronoi vertex v is equidistant from the elements of $\mathcal{P}$ whose Voronoi cells define it. Thus if $v = \bigcap_{i=0}^{d} V(p_i)$, then the p_i all lie on a common hyper sphere centred at v. For a random set of points $\mathcal{P} \subset \Re^d$, the chances of more than $d+1$ points lying on a common hyper sphere is vanishingly small [7]. The set $\mathcal{P}$ is said to be in general position if the intersection of more than $d+1$ Voronoi cells is always empty.

2.2 Delaunay triangulation.

For $\mathcal{P}$ in general position, the Delaunay triangulation of $\mathcal{P} \in \Re^d$ is the dual of the Voronoi diagram of $\mathcal{P} \in \Re^d$. In the planar setting, the duality relationship is as follows: to each Voronoi vertex c we associate a Delaunay triangle, t whose vertices are the three samples which define c. An edge $e = [p, q]$ of t is dual to the Voronoi edge $V(p) \cap V(q)$. The vertices of the Delaunay triangulation are the sample points, and they are dual to the corresponding Voronoi cells in the Voronoi diagram. Let assume triangle

Δpqr is counter clockwise and let C be the circumscribed circle of Δpqr. Consider $\mathcal{T}$ to be a triangulation of $\mathcal{P}$. Let $(\mathrm{r,u}) \in \mathcal{P}^2$ such that Δpqr and Δpqu are triangles of $\mathcal{T}$. We want to design a test which gives information about the 4th point, called u, of the polygon. If $u \in C$ then $inCircle(p,q,r,u) = 0$, else if u is outside C then $inCircle(p,q,r,u) > 0$ else u is inside the circle and satisfy $inCircle(p,q,r,u) < 0$. The inCircle test is computed based on equation 1.

$$inCircle(p,q,r,u) = \det\begin{pmatrix} 1 & p_x & p_y & p_x^2 + p_y^2 \\ 1 & q_x & q_y & q_x^2 + q_y^2 \\ 1 & r_x & r_y & r_x^2 + r_y^2 \\ 1 & u_x & u_y & u_x^2 + u_y^2 \end{pmatrix} \quad (1)$$

Given this approach, it is necessary to determine the orientation of the polygon edges in $\Re^3$. This can be establish easily by determining the orientation of tetrahedron $p\ q\ r\ u$ which is equal to orientation of $(\overrightarrow{PQ}, \overrightarrow{PR}, \overrightarrow{PU})$. Equation 2 is used to determine the orientation of the tetrahedron.

$$Orientation(p,q,r,u) = \det\begin{pmatrix} 1 & p_x & p_y & p_z \\ 1 & q_x & q_y & q_z \\ 1 & r_x & r_y & r_z \\ 1 & u_x & u_y & u_z \end{pmatrix} \quad (2)$$

All properties which will be mentioned have been treated for the planar Delaunay triangulation by exploiting the edge flip algorithm. We say that edge $e' = [r, u]$ is locally Delaunay if p is not contained in the circumscribed circle of Δqru, or equivalently, if q is not contained in the circum circle of $\Delta\ rpu$. Also this can be demonstrated by computing the sign of the inCircle (p,q,r,u) function. There is to a convenient characterization of a locally Delaunay edge e': the sum of the angles at the opposing vertices, p, and q, does not exceed π [4]. It is easy to show that if edge e is not locally Delaunay (NLD), then $p\ q\ r\ u$ is a convex quadrilateral and the opposing edge, e', will be locally Delaunay.

If $[p, q]$ is illegal, we can perform an edge flip by removing $[p, q]$ from $\mathcal{T}$ and insert $[r, u]$, see figure 1. Now $[r, u]$ becomes locally Delaunay.

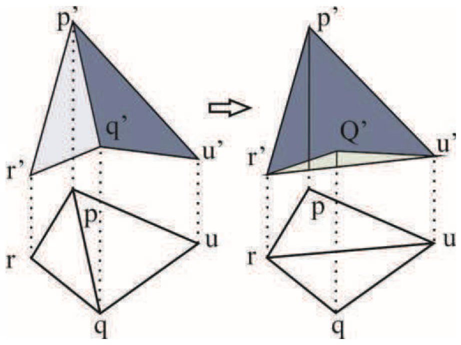

Fig. 1. *Perform an edge flip. The lifted triangulation gets lower and the upper envelope becomes convex*

Note that if all the edges in a triangulation are locally Delaunay, then the triangulation will be a Delaunay triangulation.

3 The reconstruction algorithms

Many surface reconstruction algorithms provide guarantees on the quality of the output surface if specific sampling density assumptions are met.

3.1 A first algorithm.

Consider a triangulation $\mathcal{T}$ of $\mathcal{P}$. If all edges of $\mathcal{T}$ are locally Delaunay, the algorithm ends, otherwise select an illegal edge and flip-it. The algorithm ends when all edges of $\mathcal{T}$ are locally Delaunay. The input for the algorithm is a set $\mathcal{P}$ of n planar points. The output represents the Delaunay triangulation, $\mathcal{DT}$ $(\mathcal{P})$, of those points. Figure 2 shows the algorithm in pseudo code.

```
1: compute a triangulation $\mathcal{T}$ of $\mathcal{P}$
2: initialize a stack which contain all the
   edges of $\mathcal{T}$
3: while stack has elements
4:   do pop [pq] segment from stack and
        unmark it
5:     if [pq] is illegal then
6:       do flip [pq] to [ru]
7:         for $xy \in \{\overleftrightarrow{pq}, \overrightarrow{rq}, \overrightarrow{qu}, \overrightarrow{up}\}$
8:           do if $xy$ is not marked
9:             then mark $xy$ and push it
                    on stack
10: return $\mathcal{T}$
```

Fig. 2. *The Delaunay triangulation flip test*

The program runs in $\Theta(n^2)$ time. An edge can be flipped only once because afterward it remains above the lifted triangulation. If there are $\Theta(n^2)$ edges, the algorithm will run in $\Theta(n^2)$ time.

3.1 Randomize incremental algorithm

Considering the above algorithm let suppose that we have to introduce new points in a desired area. Those points will rip the edges in that zone. In this case we have to recompute the triangles from that area. Before we introduce the new point we have to be sure that all the triangles are $\mathcal{DT}$ $(\mathcal{P})$. After that, we introduce the new point p_1 and split the surrounding triangle in 3 triangles. Perform edge flips until no illegal edge remains. After that, all the triangles become $\mathcal{DT}$. Repeat the process for all new points. Figure 3 shows the algorithm in pseudo code.

```
1: Find the triangle $\Delta pqr$ of the
     $\mathcal{DT}(\mathcal{P} \cup \{p_1\})$ containing $p_1$
2: Insert edges $\overrightarrow{p_1p}, \overrightarrow{p_1q}, \overrightarrow{p_1r}$
3: Check for conflict $\Rightarrow$ perform
   SwapTest([pq])
4:     if [pq] is an edge of the exterior
         face
5:       do return
6:       $u \leftarrow$ vertex of right edge of [pq]
7:         if inCircle($p_1$,p,q, u) < 0
8:           do flip edge [pq] for [$p_1u$]
9:     SwapTest([pu])
10:    SwapTest([uq])
11:  Perform SwapTest([uq])
12:    if [uq] is an edge of the exterior
         face
13:      do return
14:      $m \leftarrow$ vertex of right edge of [uq]
15:        if inCircle($p_1$,q,u,m) < 0
16:          do flip edge [uq] for [$p_1m$]
17:    SwapTest([qm])
18:    SwapTest([mu])
19:  Perform SwapTest([pu])
20:    if [pu] is an edge of the exterior
         face
21:      do return
22:      $n \leftarrow$ vertex of right edge of [pu]
23:        if inCircle($p_1$,p,u,n) < 0
24:          do flip edge [pu] for [$p_1n$]
25:    SwapTest([pn])
26:    SwapTest([nu])
27: SwapTest([qr]) (similar to point 3)
28: SwapTest([pr]) (similar to point 3)
```

Fig. 3. *Incremental Algorithm*

An edge between two triangles that do not contain the new point p_1 was locally Delaunay before the insertion and will remain locally Delaunay. In conclusion we flipped only edges of triangles that contain point p_1. The time needed to update the current triangulation is proportional with the number of edges that contain p_1. Each new edge will contain, after splitting the original triangle, the new point with a probability of $\frac{2}{i}$ where i is the number of the inserted points. There are $\Theta(i)$ new edges in the whole triangulation. Overall, the time needed to recompute all triangulation can be approximated using the next formula:

$$\Theta(\sum_{i=1}^{n}\frac{n}{i}) = \Theta(n\log n) \tag{3}$$

Knowing the Delaunay triangulation $\mathcal{P}$ we can find the Voronoi diagrams of $\mathcal{P}$ in $\Theta(n)$ time.

4. Aplicability

Most results pertaining to surface representation by Delaunay structures have arisen in the context of surface meshing and surface reconstruction. Both depend on surface sampling theory and the geometric accuracy of triangle meshes.

Also, the Delaunay triangulation have applicability on $\Re^2$, in generation of the nearest neighbourhood graph, minima, spanning tree (MST), or finding the largest empty circle.

In $\Re^3$ is used for 3D reconstruction, meshing, remeshing or path planning.

In surface meshing, it is used to produce a set of samples *P* and a mesh *M*, from a surface *S*. The vertices of the mesh are represented by the $\mathcal{P}$ samples. The algorithm must produce a mesh that meets given geometric accuracy requirements.

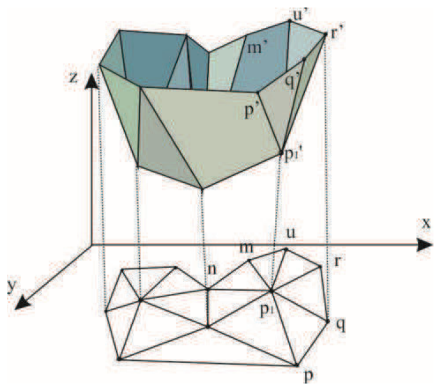

Fig. 4. *Lifting the $\mathcal{DT}(\mathcal{P})$ to produce a tree dimensional shape.*

In surface reconstruction, the input is the set of samples $\mathcal{P}$ and, aside from some regularity assumptions (i.e. that it was a smooth surface), the surface *S* is unknown. Again one wishes to construct a model that adequately represents *S* (see Figure 4).

The time needed to obtain the triangulation from Figure 4 was 30,98 milliseconds. For one point, the algorithm was covered in 2,38 milliseconds. For a complex set of point, the triangulation can be computed in seconds, thus making this approach suitable for many 3D applications.

Conclusion

In this paper we present a technique for constructing Voronoi diagrams and how to optimally compute a Delaunay triangulation. The algorithms are simples, easy to implement and efficient. The theoretical worst-case running time is $\Theta(n\log n)$, thus making this approach suitable for many real-time 3D reconstruction application. The question of space and time complexities of Delaunay refinement algorithms for three dimensional domains remains mostly open.

Acknowledgement

This paper is supported by the Sectoral Operational Programme Human Resources Development (SOP HRD), financed from the European Social Fund and by the Romanian Government under the contracts numbers, POSDRU/107/1.5/S/76945, POSDRU/89/1.5/S/59323 and POSDRU/ 88/1.5/S/59321.

References

1. Amenta, N., Bern, M., Kamvysselis, M.: *A New Voronoi- Based Surface Reconstruction Algorithm*. In: Proc. of the 25th Annual Conf. on Computer Graphics and Interactive Techniques (1998), Vol. , NY, USA, p. 415-421.
2. Amenta, N., Choi, S., Rote, G.: *Incremental constructions con BRIO*. In: Proc. of the 19th Conference on Computational Geometry (2003), NY, USA, p. 211-219.
3. Banfort, T., Sturm, P.: *Voxel Carving for Specular Surfaces*. In: Proc. Of the 19th IEEE International Conference on Computer Vision (2003), Vol.15, p.591-596.
4. Bobenko, A., Springborn, B.: *A discrete Laplace-Beltrami operator for simplicial surfaces*. In: Discrete and Computational Geometry (2007), Vol.37 (4), p. 740-756.
5. Cignoni, P., Montani, C., Scopigno R.: *A fast divide and conquer Delaunay triangulation algorithm in* E^d. In: Computer Aided Design (1998), Pisa, Italy, Vol. 30, p. 333-341.
6. Devillers, O.: *Improved incremental randomized Delaunay triangulation*. In Proc. of the 14th Annual Symp. on Comp. Geometry, NY, USA, 1998, p. 346-367.
7. Edelsbrunner, H.: *Geometry and Topology for Mesh Generation*, United Kingdom, Cambridge University Press, 2001.
8. Fuchs, H., Kedem, Z., Uselton, S.: *Optimal surface reconstruction from planar contours*. In: Comun. of the ACM (1977), Vol.20, p. 692-702.
9. Gonzalez, R., Woods, R.: *Digital Image Processing*. 2nd Edition, New Jersey, Prentice Hall, 2002.
10. Grünbaum, B., Kaibel, V., Klee, V., Ziegler, M.: *Convex polytopes*. 2nd Edition, New York & London, Springer-Verlag, 2003.
11. Guibas, L., Knuth, D., Sharir, M.: *Randomized incremental construction of Delaunay and Voronoi diagrams*. In: Algorithmica (1992), Vol. 7, p. 381-413.
12. Guibas, L., Zhang, L.: *Euclidian proximity and power diagrams*. In: Canadians Conference on Computational Geometry (1998), Canada, p. 90-91.
13. Guibas, L., Nguyen, A., Russel, D., Zhang, L.: *Collision detection for deforming necklaces*. In: Proceedings of the 18th Annual Symposium on Computational Geometry (2002), Spain, Vol. 28, p. 33-42.
14. Hartley, R., Zisserman, A.: *Multiple View Geometry in Computer Vision*. 2nd Edition, United Kingdom, Cambridge University Press, 2004.
15. Hoppe, H., et al.: *Surface reconstruction from unorganized points*. In Proc. of SIGGRAPH'96 (1996), p. 71-78.
16. Keppel, E.: *Approximating complex surfaces by triangulations of contour lines.* In: IBM Journal of Research and Development (1975), Vol. 19, p. 2-11.
17. Su, P., et al.: *A comparison of sequential Delaunay triangulation algorithms*. In Proc. of the 11th Annual Symposium on Computational Geometry (1995), NY, USA, p. 61-70.