On the Total Correctness of Lawson's Oriented Walk Algorithm

Extended Abstract

Frank Weller

Universität Dortmund
Informatik VII
44221 Dortmund
Germany
weller@ls7.informatik.uni-dortmund.de

Lawson's oriented walk is a simple algorithm for point location without preprocessing. Originally formulated for planar Delaunay triangulations [2], it generalizes immediately to tesselations of a convex, polyhedral domain into convex, polyhedral cells in d-space. Given a query point q, the algorithm works as follows.

1. Pick some cell C of the tesselation.
2. For each facet F of C, determine the position of q relative to the hyperplane that contains F.
3. If q and C lie on opposite sides of the hyperplane, step through F: If F is an interior facet of the tesselation, iterate Steps 2–4 for the cell on the other side of F. If F is a boundary facet, report that q lies outside the domain.
4. If q lies on the same side as C (or on the hyperplane) for all facets of C, report that $q \in C$.

Partial correctness of the oriented walk is easily seen. If it terminates, it locates q correctly in some cell or outside the domain. Although the algorithm is widely used, it seems that its total correctness has not been investigated. The importance of this question is illustrated by a recent publication. Mücke et al. [3] use an oriented walk in 3-dimensional Delaunay triangulations. Infinite loops can occur in this setting.

Figure 1 shows an infinitely looping oriented walk in a planar triangulation. In some triangles of the loop, there are two alternative edges through which the walk could step. The (hypothetical) implementation at hand tests the unfavorable edge first, and therefore loops. A different implementation might test the favorable edge first and terminate. If a walk happens to follow the line segment from some starting point to q, it will terminate. Thus, a given tesselation can only contain a potential infinite loop. Whether or not an oriented walk gets caught in this loop is implementation-dependent.

Proposition 1. *The oriented walk algorithm is only partially correct in planar triangulations.*

Supported by Deutsche Forschungsgemeinschaft (DFG), grant MU 744/3-2.

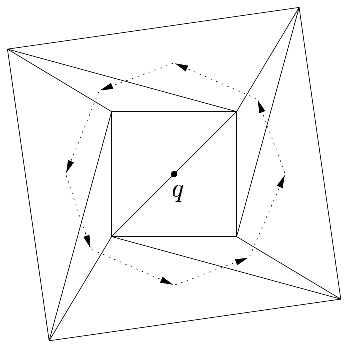

FIGURE 1. An infinitely looping oriented walk.

The example in Figure 1 can be 'inflated' to higher dimensions. Let T be a $(d-1)$-dimensional triangulation. Let S be a sufficiently large sphere in d-space, and let H be the equatorial hyperplane of S, as in Figure 2. Embed T into H such that it lies inside S. Project the vertices of T into the northern hemisphere of S, using the south pole as center of projection. The projected vertices and the south pole form the vertices of a d-dimensional triangulation T'. For each

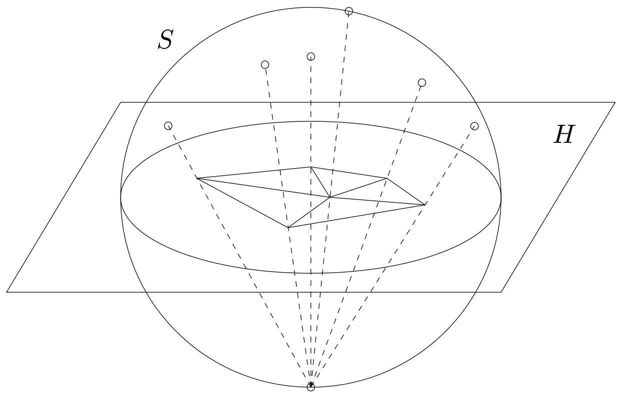

FIGURE 2. Projecting the vertices of T into the northern hemisphere.

simplex of T, let T' contain the d-dimensional simplex spanned by the corresponding projected vertices and the south pole. If necessary, add further simplices to fill the convex hull. By this construction, the facet tests in T are compatible with the corresponding facet tests in T'. If query point $q \in H$ can produce an infinite loop in T, it can produce the analogous loop in T'. Furthermore, T' is a Delaunay triangulation, since all its vertices are cospherical. By induction over d, we have

Proposition 2. *The oriented walk algorithm is only partially correct in Delaunay triangulations in dimension $d \geq 3$.*

After these negative results, let us turn to some tesselations in which the oriented walk must terminate. The first class of tesselations we consider are Voronoi diagrams. Let C and C' be two Voronoi regions which share facet F, and let p and p' be the sites that generate C and C', respectively. F is contained in the perpendicular bisecting hyperplane of p and p'. If the oriented walk steps from C to C', then q must be strictly closer to p' than to p. This strict decrease in distance from the generating site of the current cell shows that no infinite loops can occur.

Proposition 3. *The oriented walk algorithm is totally correct in Voronoi diagrams.*

In principle, the argument of decreasing distance holds for generalizations of the Voronoi diagram, too. However, only generalizations with convex, polyhedral cells admit an oriented walk in the first place. Among these are higher-order Voronoi diagrams and power diagrams. In a Voronoi diagram of order k, the generator of a region is a set of k sites. The decreasing quantity is the maximal distance from q to the sites generating the current region. In power diagrams (see, e.g., [1],) the sites are spheres. Let S be a sphere with center m and radius r. The power of a point p with respect to S is given by $\|p - m\|^2 - r^2$. This is used as a distance function to define the diagram. As the oriented walk steps from one power region to another, the power of q with respect to the current generating sphere decreases.

Proposition 4. *The oriented walk algorithm is totally correct in higher-order Voronoi diagrams and in power diagrams.*

The Delaunay tesselation of a point set M is the dual of the Voronoi diagram of M. If a Delaunay tesselation consists entirely of simplicial cells, it is the unique Delaunay triangulation of M. If M contains subsets of more than $d + 1$ cospherical points, the Delaunay tesselation may have non-simplicial cells. By splitting these cells into simplices, one obtains a non-unique Delaunay triangulation. The vertices of a Delaunay cell lie on a common sphere, which is called a Delaunay sphere. There exists a close relationship between Delaunay tesselations and power diagrams.

Proposition 5. *Let M be a finite set in d-space. In the interior of the convex hull* $\mathrm{CH}(M)$*, the Delaunay tesselation of M is identical to the power diagram of the Delaunay spheres of M.*
(Proof omitted due to space limitations.)

Corollary 6. *The oriented walk algorithm is totally correct in Delaunay tesselations.*

If an infinite loop occurs in a Delaunay triangulation, this loop must be contained in a single Delaunay cell (which has been split into several simplices.) This is also the case for the 'inflated' triangulations that prove Proposition 2. In fact, each of those triangulations is contained in a single Delaunay cell. Mücke et al. ([3], see above) tested their algorithms on sets of random points. It is not surprising that no infinite loops occured. In engineering applications, such as FEM mesh generation, one is more likely to encounter large sets of cospherical points.

We have yet to look at the original scope of the oriented walk algorithm, planar Delaunay triangulations. A planar Delaunay cell is a convex polygon. The edges which split it into triangles are diagonals. This precludes infinite loops within one Delaunay cell: An oriented walk would have to leave a triangle through a diagonal of the cell and later come back through the same diagonal in reverse direction.

Proposition 7. *The oriented walk algorithm is totally correct in planar Delaunay triangulations.*

Remark 8. All results were derived under the assumption of exact computation, neglecting possible effects of rounding errors. Palacios-Velez and Cuevas Renaud [4] report infinite loops consisting of two adjacent triangles due to numerical problems.

References

[1] Franz Aurenhammer. Power diagrams: Properties, algorithms and applications. *SIAM Journal on Computing*, 16(1):1171–1181, February 1987.

[2] C. L. Lawson. Software for C^1 surface interpolation. In J. R. Rice, editor, *Mathematical Software III*, pages 161–194. Academic Press, New York, 1977.

[3] Ernst P. Mücke, Isaac Saias, and Binhai Zhu. Fast randomized point location without preprocessing in two- and three-dimensional Delaunay triangulations. In *Twelfth Annual Symposium on Computational Geometry, Philadelphia, PA, USA, May 24–26 1996*, pages 274–283. ACM Press, New York, 1996.

[4] O. Palacios-Velez and B. Cuevas Renaud. A dynamic hierarchical subdivision algorithm for computing Delaunay triangulations and other closest-point problems. *ACM Transactions on Mathematical Software*, 16(3):275–292, September 1990.