Kurt E. Brassel and
*Douglas Reif**

A Procedure to Generate Thiessen Polygons

An algorithm to generate Thiessen diagrams for a set of n points defined in the plane is presented. First, existing proximal polygon computation procedures are reviewed and terms are defined. The algorithm developed here uses a rectangular window within which the Thiessen diagram is defined. The computation of Thiessen polygons uses an iterative walking process whereby the processing starts at the lower left corner of the diagram and proceeds toward the right top corner. The use of a sorted point sequence and dynamical core allocation provide for efficient processing. The presentation is concluded by the discussion of an implementation of the algorithm in a FORTRAN *program.*

1. INTRODUCTION

In 1911 the climatologist A. H. Thiessen suggested a new method of representing precipitation data from unevenly distributed weather stations. He defined regions based on a set of data points in the plane (weather stations) such that "regions be enclosed by a line midway between the station under consideration and surrounding stations" [*36*, p. 1083]. Based on this proposal the term *Thiessen polygon* has since been commonly used in geography to denote polygons defined by proximity criteria with respect to a set of points in the plane. The net of all Thiessen polygons defined by the point set is called a

*Some preliminary studies leading to this algorithm have been initiated by Dr. Thomas Peucker, Simon Fraser University, and have been supported by ONR Contract N00014-75-0886. Mr. Robert Fowler, Simon Fraser University, has pointed out and corrected some important shortcomings of a preliminary version of this algorithm. The implementation of the present procedure has been supported by the Erie and Niagara Counties Regional Planning Board. Mr. George Sicherman reviewed the manuscript and provided substantial help in formulating the proof section of this paper. Mr. Mike Wasilenko executed the illustrations. The authors gratefully acknowledge all these contributions.

Kurt E. Brassel is assistant professor of geography and Douglas Reif is a graduate student in computer science, State University of New York at Buffalo.

0016-7363/79/0779-0289$00.50/0 © 1979 Ohio State University Press
GEOGRAPHICAL ANALYSIS, vol. 11, no. 3 (July 1979)
Submitted 6/78. Revised version accepted 12/78.

Thiessen diagram, or alternatively a Voronoi diagram [*37*], Wigner-Seitz cells, or Dirichlet tesselation. Hoey has proposed the more descriptive and impartial term "proximal polygons" [*31*, p. 196]. The dual of a Thiessen diagram is a triangulation based on proximity criteria. Delaunay [*10*] first recognized this dual relationship; therefore, the term *Delaunay triangulation* is used for the dual form of proximal diagrams.

Traditionally, proximal polygons are constructed by computing perpendicular bisectors between all neighboring points of the set. The major problem, however, is to define the "neighbors" of each point set. This problem is fundamental in geometry and has some impact to related problems such as the "post office problem" [*19*] and the construction of spanning trees. The Thiessen polygon is one of the most fundamental and useful constructs defined by irregular lattices.

A variety of scientific contributions deal with the various aspects of proximal polygon generation and use. A first group represents early introductions to the method and intuitive rules for construction [*10, 13, 36, 37, 38*].

A second group discusses the theoretical aspects of the method. Rogers [29] defines the mathematical properties of the Thiessen diagrams. Shamos [*31, 32*], Shamos and Hoey [*34*], and Shamos and Bentley [*33*] lay the theoretical framework for the computational aspects of proximal polygons. They use proximity problems as a test case and as an illustrative example for the emerging discipline of computational geometry. Computational geometry "seeks to understand the interaction between geometry and computing. Its task is to isolate fundamental problems and develop optimal algorithms to solve them, building, so to speak, a set of computing tools that can be used in diverse applications" [*33*, p. 1]. Methodologically, computational geometry concentrates on the analysis of the complexity of geometrical problems and algorithms to solve them. The complexity of a problem can be expressed by the order of magnitude of the number of operations used in solving it. For a problem involving N objects the following notation for complexity is used: "$O(f(N))$ stands for the set of all functions $g(N)$ such that there exist positive constants C and M with $|g(N)| \leqslant Cf(N)$ for all $N \geqslant M$" [*33*, p. 2]. A problem can be said to require at least $O(f(N))$ operations to be solved (lower bound), where a specific algorithm may solve this problem in the worst case in $O(f(N))$ operations (upper bound).

Shamos, Bentley, and Hoey establish several theorems concerning the complexity of various nearest neighbor tasks. The major findings include the following: (a) "$O(N \log N)$ is a lower bound on the time required to determine the two closest of N points in dimension one and higher" [*34*, p. 152]; (b) "$O(N \log N)$ is a lower bound on the time required to triangulate N points in dimension two or higher" [*34*, p. 153]; (c) "the Voronoi diagram of N points in the plane can be constructed in $O(N \log N)$ time" [*34*, p. 151]; and (d), "given the Voronoi diagram on N points in the plane, their convex hull can be found in linear time" [*31*, p. 209]. Shamos and Hoey [*34*] also generalize the proximal diagram by defining regions dealing with the farthest points, the k closest points, and so on. Bentley and Shamos [*1, 2*] further extend the analysis of closest point problems to multidimensional space. Lloyd [22] demonstrates that a Delaunay triangulation is not a minimum weight triangulation. Sibson [*35*] proves that the Delaunay triangulation is the only triangulation that is locally equiangular. Fowler [*14*] presents a comparative study of searching algorithms where the strength of proximal diagrams is demonstrated.

Other algorithms and computer procedures have been proposed for proximal map construction and related problems. Shamos and Hoey [*34*] present a divide-and-conquer algorithm, which they prove to be theoretically optimal. Fowler [*14*, *15*] developed "walking" algorithms for both the construction of, and search procedures within, Delaunay triangulations. Green and Sibson [*17*] propose a recursive algorithm and present a thorough analysis of its implementation. Other algorithms are given by Rhynsburger [*28*], Lawson [*21*], McLain [*23*], and Mead [*24*].

The applications of the proximal polygon concepts are many and varied—in locational analysis [*3*, *4*, *5*, *6*, *18*], in plant ecology [*24*], cristallography [*16*], and as a basic organizational structure in geographic information systems [*8*, *27*, *33*]. Crain [*9*] uses the concept to establish properties of Poisson distributions in two-dimensional space, and Schachter [*30*] bases the decomposition of polygons into convex sets on the Thiessen structure.

The present paper presents another algorithm to define Thiessen diagrams. It uses a "walking" algorithm and is based on a point sort and subsequent local processing. A preliminary version of this concept is described in [*7*].

2. DEFINITIONS

A set N of n points is given in the plane, which in this discussion are called *centroids* (see Fig. 1). Find a set of points V in the plane such that each $V_i \in V$ is equidistant and closest to at least three centroids; these points are termed (Thiessen) *vertices*. A (Thiessen) *edge* is the locus of all points equidistant and closest to two centroids; Thiessen edges may be delimited by two vertices or may be unlimited in one direction. A *Thiessen polygon* is defined as the locus of all points closer to a centroid $C \in N$ than to any other centroid. This definition implies that Thiessen polygons are convex. The set of n centroids determines a set of n Thiessen polygons. The set of all polygons is called a *Thiessen diagram.*

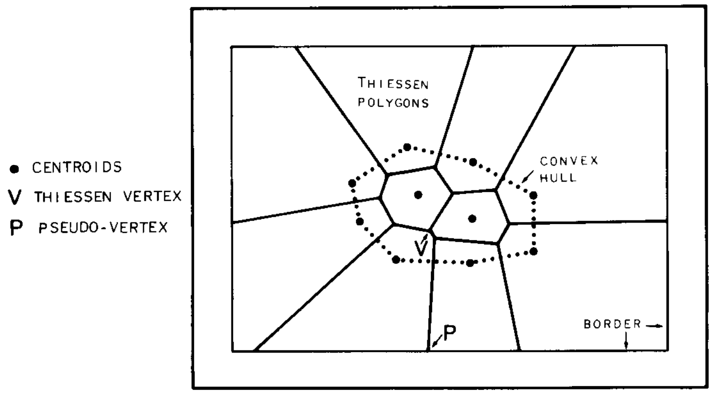

FIG. 1. Definition of Terms Used for Thiessen Polygon Computation

Thiessen polygons may be *closed* or *open*; closed polygons are entirely bounded by Thiessen edges, open polygons extend to infinity. Define the *convex hull* of the set N of centroids as the smallest convex polygon enclosing all centroids. All centroids of the boundary of the convex hull have open Thiessen polygons, and all interior centroids have closed polygons. A centroid B is called a Thiessen *neighbor* of a centroid C if the Thiessen polygons about the two centroids have an edge in common; they are called *half-neighbors* if they have a single vertex in common.

3. BASIC CONCEPTS

Assume a given set of centroids in the plane. Given also is a rectangular window that includes the set of centroids; this window is further referred to as the *border* of the Thiessen diagram. In order to ease computation, the border is selected parallel to the Cartesian axes and the centroids are sorted in x-direction. In our procedure the Thiessen diagram will be restricted to the Cartesian rectangle, where portions of the border delimit open Thiessen polygons. This delimitation of the Thiessen diagram is achieved by adding some dummy points to the point set during processing time (see Fig. 2).

Given the centroid A, four dummy points W through Z are introduced such that perpendicular bisectors between A and the dummy points generate the four edges of the border (Fig. 2). If A is the point closest to the lower left corner V_o then V_o must be a vertex of the Thiessen polygon about A, and dummy point X is a neighbor of A. Further searching can be reduced to a search for the "next neighbor of A clockwise to X." In this discussion the phrase "next clockwise neighbor" will be used for this relationship.

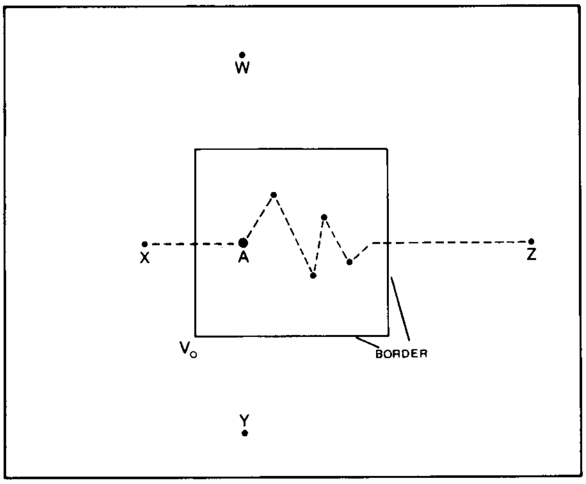

FIG. 2. Inserting Dummy Data Points to the Point Set

The procedure to search for a clockwise neighbor is based on the following assumptions:

- •Thiessen polygons are always convex.
- •The next clockwise neighbor is located in the halfplane in a clockwise direction from a vector connecting the centroid and the present neighbor. (This holds only for interior centroids. Through the introduction of the dummy points, however, all centroids of the original set can be considered interior.)
- •A vertex V equidistant to three centroids A, B, and C is a true Thiessen vertex if no other centroid is located within a circle of radius R about V (R is the distance between vertex V and the three centroids A, B, and C).

The first two assumptions are proven in the appendix; a proof for the third assumption is given by Shamos [*31*, p. 201]. The strategy used for the computation of the next neighbor of A clockwise to X is as follows:

1. Define the two halfplanes E_1, E_2: E_1 is clockwise with respect to a vector AX, E_2 is counterclockwise with respect to AX
2. Select a first potential neighbor P in the halfplane E_1. One of the dummy points will always suffice. Find the point V equidistant to A, X, and P. V is called a potential vertex and R is the distance from V to A, X, and P, respectively.
3. Search for point P', which is both in the halfplane E_1 and inside the circle (V,R). If such a point P' exists, compute (V',R') equidistant to A, X, P', where $R'<R$; declare P' the new potential neighbor. When there are no points remaining within the circle/halfplane, the potential neighbor is the next clockwise neighbor.

The above strategy is illustrated in Figures 3 and 4. Point X is a known neighbor of A. To find the next clockwise neighbor, select point W as a potential neighbor. Compute V, R by intersecting the bisectors defined by pairs of A, X, and W, and search for a point in (V,R) in E_1. Start this search at point X and follow the sorted sequence of the point set (indicated by dashed lines in Fig. 3a), and find B. Since B is within the circle, it is the new potential neighbor.

Compute V' and R' (Fig. 3b). Once it is proven that no other point is within (V',R'), declare point B to be the next clockwise neighbor and V' a vertex of the Thiessen polygon about A.

Search for the next clockwise neighbor of A with respect to B (Fig. 3c): redefine the halfplanes E_1, E_2 and use W as the first potential neighbor, compute (V,R). Find D within (V,R), compute V', R'. Since there is no further point within E_1 and (V',R'), D is the next clockwise neighbor and V' is a polygon vertex. The detailed algorithm for the search for the next clockwise neighbor is given in Figure 4. In special cases—as illustrated in Figure 5a—a point P may be situated on the perimeter of the circle defined by A, P, X. This implies that A, P, and X, P' are half-neighbors. The procedure retains the "less clockwise" of the two points P and P', and the other is considered outside the circle. In Figure 5b, A, P are recorded as (full) neighbors and X, P' are not considered neighbors at all. V_1, V_2 are vertices with identical physical locations.

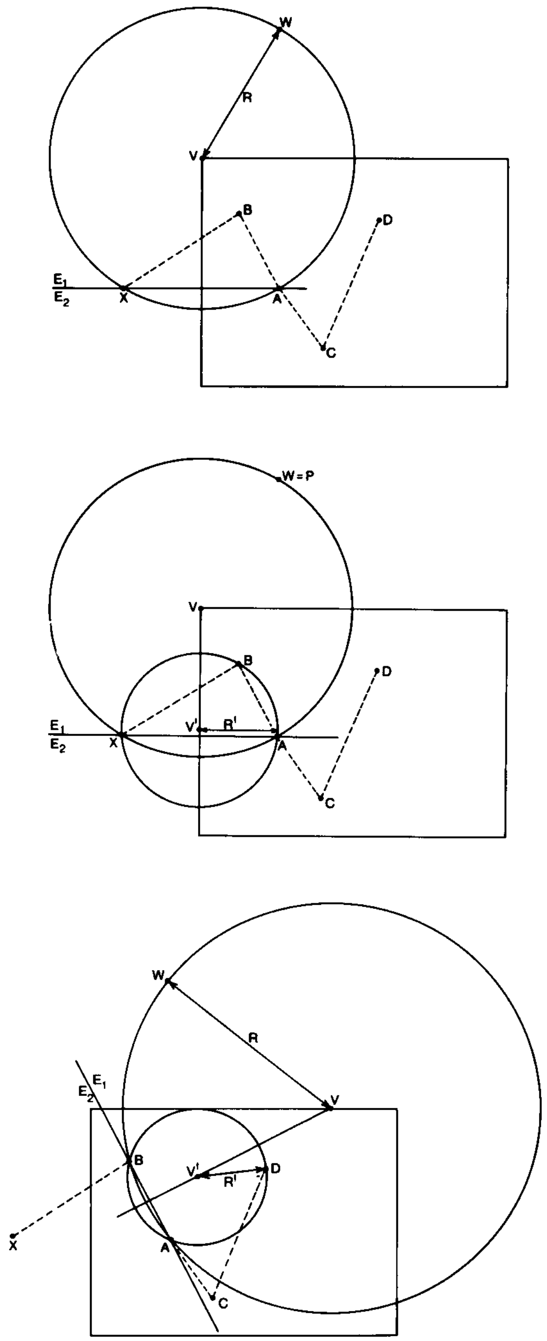

FIG. 3. Strategy for the Computation of the Next Neighbor of *A* Clockwise to *X*

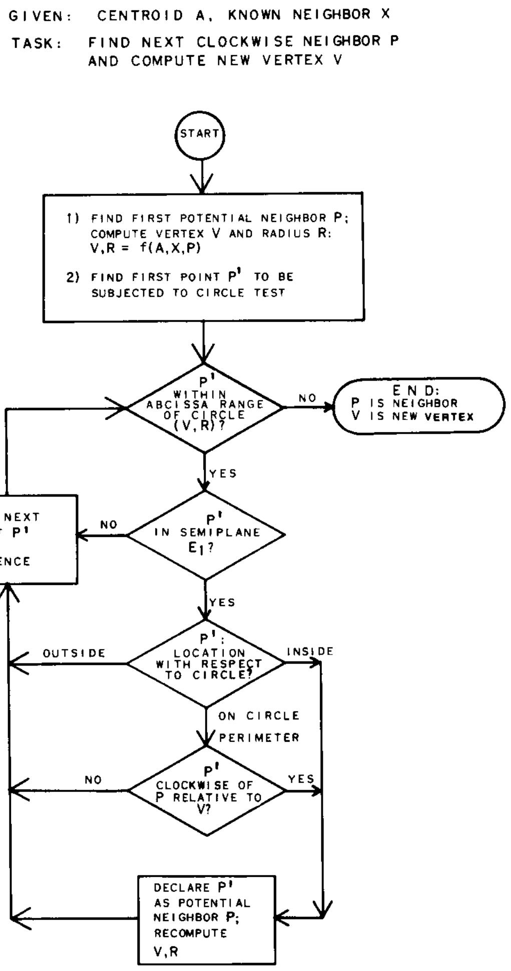
GIVEN: CENTROID A, KNOWN NEIGHBOR X
TASK: FIND NEXT CLOCKWISE NEIGHBOR P AND COMPUTE NEW VERTEX V
START
1) FIND FIRST POTENTIAL NEIGHBOR P; COMPUTE VERTEX V AND RADIUS R: V,R = f(A,X,P)
2) FIND FIRST POINT P' TO BE SUBJECTED TO CIRCLE TEST
P' WITHIN ABCISSA RANGE OF CIRCLE (V,R)?
NO
END: P IS NEIGHBOR V IS NEW VERTEX
YES
P' IN SEMIPLANE E1?
NO
FIND NEXT POINT P' IN SEQUENCE
YES
P': LOCATION WITH RESPECT TO CIRCLE?
OUTSIDE
INSIDE
ON CIRCLE PERIMETER
P' CLOCKWISE OF P RELATIVE TO V?
NO
YES
DECLARE P' AS POTENTIAL NEIGHBOR P; RECOMPUTE V,R

Fig. 4. Procedure to Find Next Clockwise Neighbor (Circle Test)

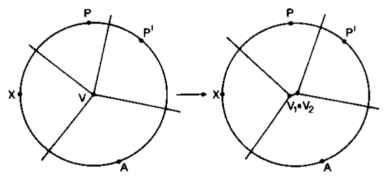

Fig. 5. Handling of the Halfneighbor Problem

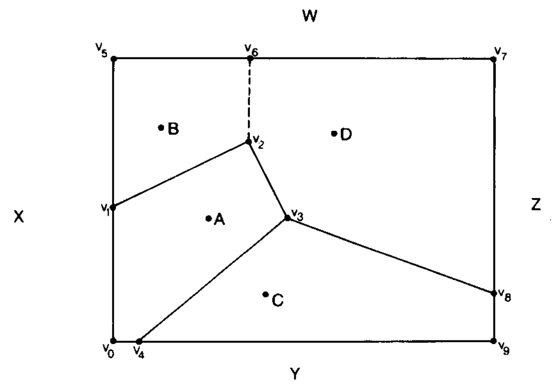

FIG. 6. **Illustration Example for the Processing Sequence**

The search for next clockwise neighbors is repeated until the polygon about the centroid in question is completed, and all the neighbors are recorded in a neighborhood record. For the polygon about *A* in Figure 6, the neighborhood recorded is as follows: *A*: *X*, *B*, *D*, *C*, *Y*, (*X*).

Since each vertex V_i pertains to three Thiessen polygons, all neighborhood relationships are mutual. Once a vertex is found it is stored and the mutual relationships among the centroids involved are recorded in a bookkeeping process. Whenever a new vertex V_i in Figure 6 is found, the following entries are made into the neighborhood file (modifications to the neighborhood file in each step are in boldface).

V_0:	*A*:	**Y, X** (for the dummy points no neighborhood records are kept)
V_1:	*A*:	*Y*, *X*, **B**
	B:	**A, X**
V_2:	*A*:	*Y*, *X*, *B*, **D**
	B:	**D**, *A*, *X*
	D:	**A, B**
V_3:	*A*:	*Y*, *X*, *B*, *D*, **C**
	B:	*D*, *A*, *X*
	D:	**C**, *A*, *B*
	C:	**A, D**
V_4:	*A*:	*Y*, *X*, *B*, *D*, *C*, (**Y**) *completed*
	B:	*D*, *A*, *X*
	D:	*C*, *A*, *B*
	C:	**Y**, *A*, *D*

Since the polygon about *A* is now completed, its neighbor record is eliminated from the list, and the next centroid is chosen from the list for processing. This is centroid *B* with *D*, *A*, *X* as known neighbors, where *X* is the most clockwise point. The four dummy points are shifted to a position such that the bisectors between dummy points and centroid *B* generate the four edges of the border.

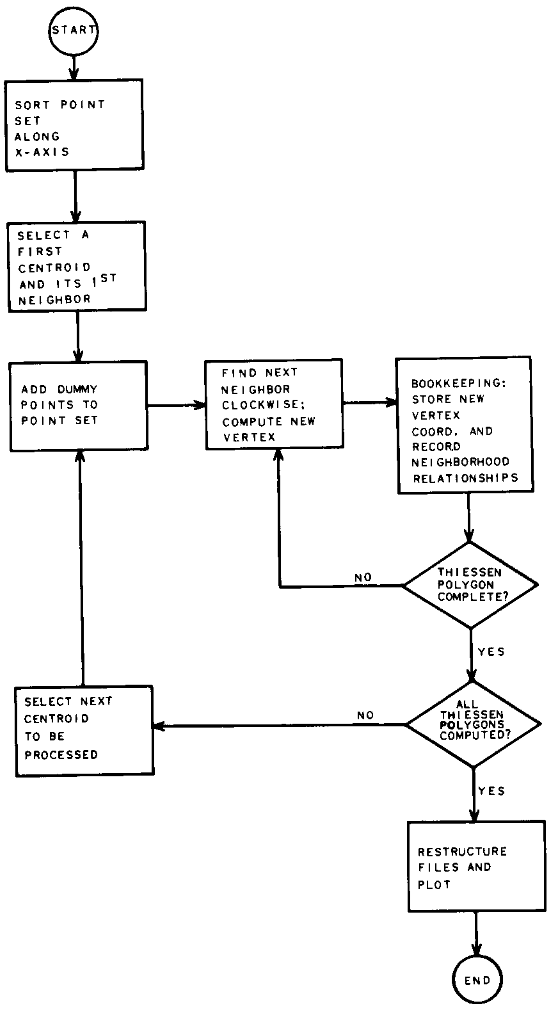
START
SORT POINT SET ALONG X-AXIS
SELECT A FIRST CENTROID AND ITS 1ST NEIGHBOR
ADD DUMMY POINTS TO POINT SET
FIND NEXT NEIGHBOR CLOCKWISE; COMPUTE NEW VERTEX
BOOKKEEPING: STORE NEW VERTEX COORD. AND RECORD NEIGHBORHOOD RELATIONSHIPS
THIESSEN POLYGON COMPLETE?
NO
YES
ALL THIESSEN POLYGONS COMPUTED?
NO
YES
SELECT NEXT CENTROID TO BE PROCESSED
RESTRUCTURE FILES AND PLOT
END

FIG. 7. The Framework of the Algorithm

Processing continues:

V_5:	B:	$D, A, X,$ **W**
	D:	C, A, B
	C:	Y, A, D
V_6:	B:	$D, A, X, W,$ (**D**) *completed*
	D:	$C, A, B,$ **W**
	C:	Y, A, D

Remove *B* and process *D*:

V_7: *D*: *C*, *A*, *B*, *W*, Z
C: *Y*, *A*, *D*
V_8: *D*: *C*, *A*, *B*, *W*, Z, (C) *completed*
C: *Y*, *A*, *D*, Z
V_9: *C*: *Y*, *A*, *D*, Z, (Y) *completed*

Since the list is exhausted at this point, the Thiessen diagram is completed. Through the use of the dynamically modified list of neighbor records space requirements are kept modest.

In summary, the computation of Thiessen polygons begins with the centroid closest to the lower left-hand corner of the window and proceeds towards the centroids near the right top. The search for neighbors and the computation of vertices for a single polygon is performed in a search clockwise around the centroid. Once the entire diagram is completed the information pertaining to vertices and neighborhood relations is restructured and the diagram is plotted. The overall framework of the procedure is outlined in Figure 7.

4. ALGORITHM IMPLEMENTATION AND EVALUATION

The algorithm as described has been encoded in Fortran IV and test runs have been performed on a CDC Cyber 173 (Fig. 8). The procedure is constructed of four overlays performing such tasks as input and sort, Thiessen computation, merging of polygon descriptor records with vertex and name files, and plot.

The program as implemented allows for processing of Thiessen diagrams with up to 5,000 points. This implementation requires 46000_{10} (107000_8) words on a CDC Cyber 173. A series of test computations has been run for Thiessen

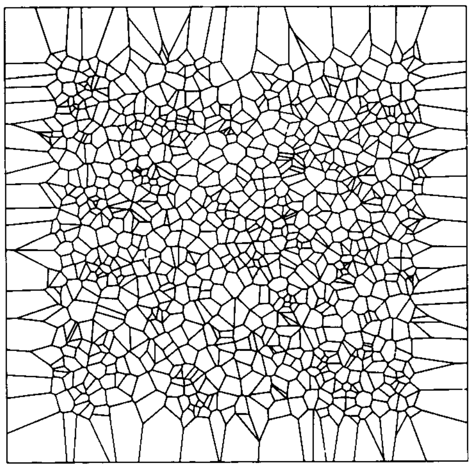

FIG. 8. Thiessen Test Example: 1,000 Points

diagrams with 20–5,000 randomly distributed data points. Other test runs included data sets with one or two centroid clusters. The cumulative processing time for the several steps of the Thiessen procedure is shown in Figure 9. Recognize the near linear characteristics of these curves and the slight reduction in processing time for clustered data sets. Figure 10 shows the observed average computation time for each Thiessen diagram as a function of the number of data points in semilogarithmic scale.

Further analysis of the Thiessen procedure is shown in Figure 11. It represents empirically computed counts of processing steps. The top curve (A) indicates the average number of data points consulted in order to create a Thiessen polygon. These data points are subjected to the test for overall rejection ("is P' within abcissa range of circle (V, R)?) and the diagonal test ("is P' in the halfplane E_1?"). These two tests include a total of five additions, three multiplications, and two arithmetic branching checks. Approximately 50 percent of the points pass these tests; they are represented in curve B and are checked for their location within the circle about the potential vertex. This test includes two adds, one multiplication, and two arithmetic checks. As may be seen from curve C, it further reduces the number of data points to be considered drastically. For all points inside the circle (curve C) Thiessen vertices have to be computed in a procedure including twenty-six additions and seventeen multiplications. From this figure it is evident that many points are subjected to simple tests, whereas the time-consuming processes are performed only for relatively few centroids. This figure also indicates that the number of processing steps is lower for clustered data sets.

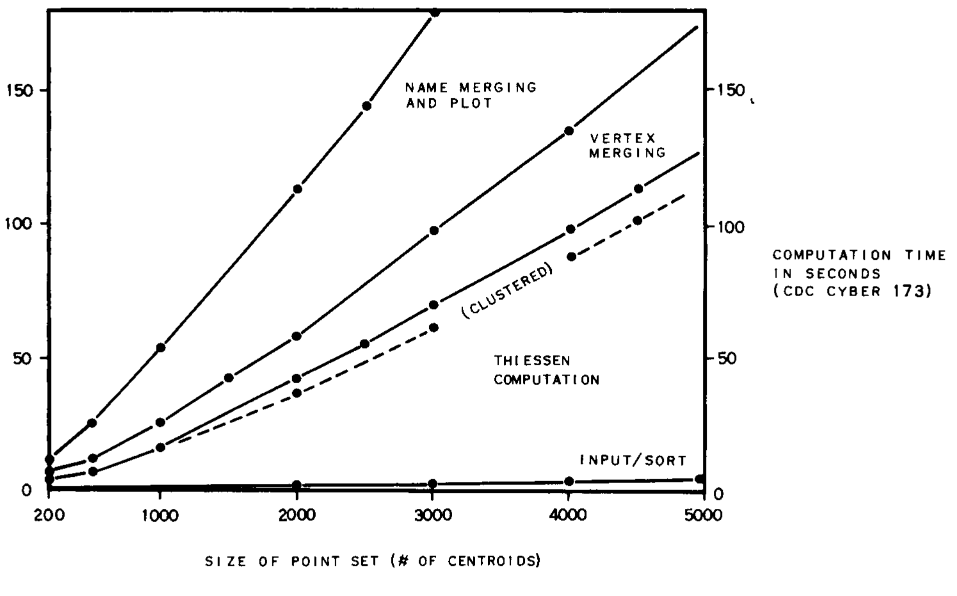

FIG. 9. Cumulative Processing Time for Thiessen Runs

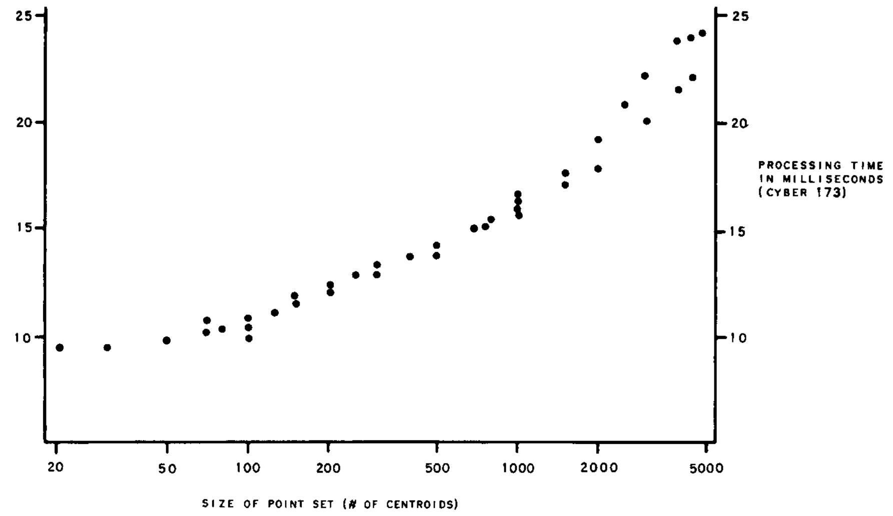

FIG. 10. Average Computation Time for One Thiessen Polygon as a Function of the Number of Data Points on the Point Set (in milliseconds, CDC Cyber 173).

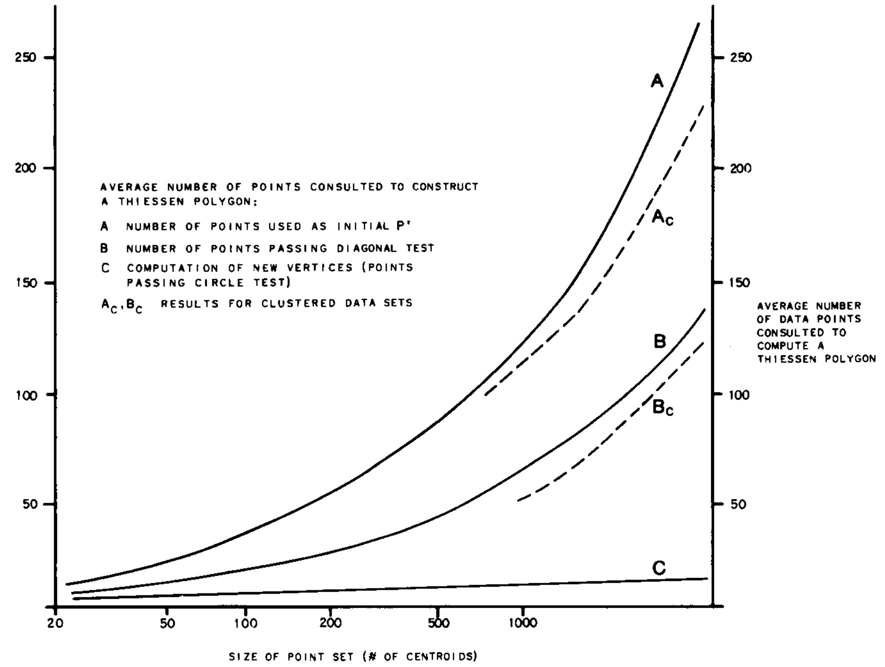

FIG. 11. Empirically Computed Counts of Processing Steps

6. CONCLUSIONS

This paper has discussed the terminology and traditional approaches of Thiessen diagrams and introduced a new algorithm to create Thiessen polygons. The algorithm uses an iterative approximation procedure to find Thiessen adjacencies. It makes use of a window to simplify the procedure. Testing with data sets of up to 5,000 points has proven the procedure to be efficient with empirical results on the order of $4.5 \times N \times N^{-5}$ milliseconds for a Thiessen diagram of N points. The strength of the proximal polygon construct and the range of applications have been pointed out in the introductory section. Of particular interest to the authors is the use of proximity-based triangulations as storage structure for geographic information systems. Their use for complex mapping tasks will be a further subject of research.

APPENDIX: PROOFS FOR BASIC ASSUMPTIONS USED FOR THIESSEN COMPUTATION

LEMMA 1. *Thiessen polygons are always convex.*

SUBLEMMA. *The intersection of any number of convex sets is convex.*

Proof. Let $S_i, i \in I$, be convex, and let $S = \cap S_i$. If a and b are two points in S, then they are in each S_i, so the line joining a to b lies in each S_i, and therefore in S.

Proof of main lemma: The locus of points closer to a centroid A than another centroid B_i is a halfplane H_i. The Thiessen polygon about A is the intersection of the halfplanes H_i for all $B_i \neq A$ in the data set. But every halfplane is convex; therefore, the Thiessen polygon about A is convex.

LEMMA 2. *Given a centroid C, which is known to create a closed Thiessen polygon, assume a radial coordinate system in which clockwise angles are positive* (*Fig.* 12). *Let point P be a true Thiessen neighbor of C. The straight line CP subdivides the plane into two halfplanes E_1 and E_2. Lemma*: *For a closed Thiessen polygon the true next neighbor about C clockwise of point P must lie in the halfplane E_1.*

Proof. Assume at least one point P' in the halfplane E_1 (closed Thiessen polygon). The angular relationship between the vector CP and the direction of

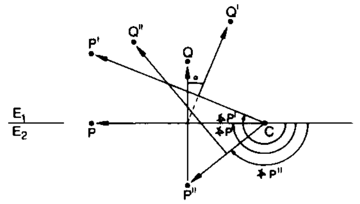

FIG. 12. Angular Relationship between Vectors Connecting Neighbors and Bisectors

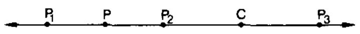

FIG. 13. Points on the Straight Line through *CP*

the vector of the perpendicular bisector Q can be expressed as

$$\sphericalangle Q = \sphericalangle P + 90°.$$

Likewise, for the next neighbor, P' follows that $\sphericalangle Q' = \sphericalangle P' + 90°$, the angle between directions Q and Q' is therefore

$$\alpha = \sphericalangle Q' - Q = \sphericalangle P' - P.$$

For all $\sphericalangle P'' < \sphericalangle P$ the angle between Q and Q'' will be negative ($\sphericalangle Q_2'' - \sphericalangle Q < 0$). Since Q'' is the direction of an edge subsequent to the edge Q in clockwise order, the above condition implies that the Thiessen polygon about a point C is not convex, which has been proven impossible. All points P'' in the halfplane E_2 can therefore be excluded as candidates for the next neighbor in clockwise direction of a closed Thiessen polygon. Likewise, points on the straight line through CP can be excluded (Fig. 13): P_1 cannot be a neighbor of C if P exists, P_2 cannot exist if P is known to be a neighbor of C, and P_3 would create an open Thiessen polygon.

LITERATURE CITED

1. Bentley, J. L. "Divide and Conquer Algorithms for Closest Point Problems in Multidimensional Space." Ph.D. thesis, University of North Carolina at Chapel Hill, 1976.
2. Bentley, J. L., and M. I. Shamos. "Divide and Conquer in Multidimensional Space." *Proceedings of the Eighth Annual ACM Symposium on Theory of Computing*, Hershey, Pa., 1976, pp. 220–30.
3. Besag, J. "Spatial Interaction and the Statistical Analysis of Lattice Systems." *Journal of the Royal Statistical Society* (*B*), 36 (1974), 192–236.
4. Boots, B. M. "Contact Number Properties in the Study of Cellular Networks." *Geographical Analysis*, 9 (1977), 379–87.
5. ______. "Delaunay Triangles: An Alternative Approach to Point Pattern Analysis." *Proceedings of the Association of American Geographers*, 6 (1974), 26–29.
6. ______. "Some Models of the Random Subdivision of Space." *Geografiska Annaler*, Ser. B, 55 (1973), 34–48.
7. Brassel, K. E. "Neighborhood Computations for Large Sets of Data Points." *Proceedings of the International Symposium on Computer-Assisted Cartography* (*AUTO-CARTO II*). Washington, D.C.: American Congress on Surveying and Mapping, 1976, 337–45.
8______. "A Topological Data Structure for Multi-Element Map Processing." *Harvard Papers on Geographic Data Structures*, 4 (1978).
9. Crain, I. K. "The Monte Carlo Generation of Random Polygons." *Computers and Geoscience*, 4 (1978), 131–41.
10. Delaunay, B. "Sur la sphere vide." *Bulletin of the Academy of Sciences of the USSR*, Classe Sci. Mat. Nat. (1934), 793–800.
11. Dobkin, D. P., and R. J. Lipton. "The Complexity of Searching Lines in the Plane." Department of Computer Science, Yale University, Research Report No. 71, 1975.

12. ______. "Multidimensional Searching Problems." *SIAM Journal of Computing*, 5 (1976), 181–86.
13. Fisher, H. "Instructions for Establishing Proximal Zones." Unpublished manuscript. Harvard University, 1970.
14. Fowler, R. J. "Approaches in Multi-Dimensional Searching." *Harvard Papers in Geographical Information Systems*, 6 (1978).
15. ______. "DELTRI: An Efficient Program for Producing Delaunay Triangulations." Technical Report 18, ONR Contract N00014-75-0886, Department of Geography, Simon Fraser University, 1977.
16. Gilbert, E. N. "Random Subdivisions of Space into Crystals." *Annals of Mathematical Statistics*, 33 (1962), 958–72.
17. Green, P. J., and R. Sibson. "Computing Dirichlet Tesselations in the Plane." *Computer Journal*, 21 (1978), 168–73.
18. Keeney, R. L. "A Method for Districting among Facilities." *Operations Research*, 20 (1972), 613–18.
19. Knuth, D. E. *The Art of Computer Programming, Vol.* 3: *Sorting and Searching*. New York: Addison Wesley, 1973.
20. Laboratory for Computer Graphics and Spatial Analysis. *SYMAP User's Reference Manual*. Cambridge: Harvard University, 1975.
21. Lawson, C. L. "Generation of a Triangular Grid with Application to Contour Plotting." California Institute of Technology, Jet Propulsion Laboratory, Technical Memorandum No. 229, 1972.
22. Lloyd, E. L. "On Triangulations of a Set of Points in the Plane." Master's thesis, MIT, 1977.
23. McLain, D. H. "Two Dimensional Interpolation from Random Data." *Computer Journal*, 19 (1976), 178–81.
24. Mead, R. "Models for Interplant Competition in Irregularly Spaced Populations." *Statistical Ecology*, 2 (1971), 13–30.
25. Nagy, G., and S. Wagle. "Geographic Data Processing." Department of Computer Science, University of Nebraska, 1978.
26. Peucker, T. K., and N. Chrisman. "Cartographic Data Structures." *The American Cartographer*, 2 (1975), 55–69.
27. Peucker, T. K., R. J. Fowler, J. J. Little, and D. M. Mark. "The Triangulated Irregular Network." *Proceedings of the Digital Terrain Models* (*DTM*) *Symposium*, American Society of Photogrammetry, St. Louis, 1978, 516–40.
28. Rhynsburger, D. "Analytic Delineation of Theissen Polygons." *Geographical Analysis*, 5 (1973), 133–44.
29. Rogers, C. A. *Packing and Covering*. London: Cambridge University Press, 1964.
30. Schachter, B. "Decomposition of Polygons into Convex Sets." *IEEE Transactions on Computers*, C-27 (1978), 1078–82.
31. Shamos, M. I. "Computational Geometry." Ph.D. thesis, Yale University, 1977.
32. ______. "Geometric Complexity." *Proceedings of the Seventh ACM Symposium on the Theory of Computing* (1975), 224–33.
33. Shamos, M. I., and J. L. Bentley. "Optimal Algorithms for Structuring Geographic Data." *Harvard Papers on Geographical Information Systems*, 6 (1978).
34. Shamos, M. I., and D. Hoey. "Closest Point Problems." *Proceedings of the Sixteenth Annual Symposium on the Foundations of Computer Science*, IEEE (1975), 151–62.
35. Sibson, R. "Locally Equiangular Triangulations." Unpublished manuscript.
36. Thiessen, A. J., and J. C. Alter. "Precipitation Averages for Large Areas." *Monthly Weather Review*, 39 (1911), 1082–84.
37. Voronoi, G. "Nouvelles applications des parametres continus à la theorie des formes quadratiques, Deuxieme Memoire, Recherches sur les paralleloedres primitifs." *Journal Reine Angew. Math.*, 134 (1908), 198–287.
38. Whitney, E. M. "Areal Rainfall Estimates." *Monthly Weather Review*, 57 (1929), 462–63.
39. Wirth, N. *Algorithms + Data Structures = Programs*. Englewood Cliffs: Prentice-Hall, 1975.