INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

UNIVERSITY OF ALBERTA

Spatial Tessellations of Wildlife Distributions

By

Timothy Trent Quinn ©

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Master of Science

in

Wildlife Ecology and Management

DEPARTMENT OF RENEWABLE RESOURCES

Edmonton, Alberta
Fall 1997

National Library of Canada

Acquisitions and Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale du Canada

Acquisitions et services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-22660-3

Canada

University of Alberta

Library Release Form

Name of Author: Timothy Trent Quinn.

Title of Thesis: Spatial Tessellations of Wildlife Distributions.

Degree: Master of Science.

Year this Degree Granted: 1997.

Permission is hereby granted to the University of Alberta to reproduce single copies of the thesis and to lend or sell such copies for private, scholarly, or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright of the thesis, and except as hereinbefore provided, neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.

[illegible]

11332-73rd Avenue.
Edmonton, Alberta, Canada.
T6G-0C8.

October 3, 1997.

UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled SPATIAL TESSELLATIONS OF WILDLIFE DISTRIBUTIONS submitted by Timothy Trent Quinn in partial fulfillment of the requirements for the degree of MASTER OF SCIENCE in WILDLIFE ECOLOGY AND MANAGEMENT.

[illegible]
J.A. Beck

[illegible]
R.J. Hudson

[illegible]
J. Hodgson

[illegible]
J. Culberson

Date: Oct 3, 1997

I dedicate this thesis to my wife, Anne,

for her inspiration, love and advice,

to my mother, Maxine,

for making me learn to fly on my own,

and to the cat,

Jasmine Bobo Hoot Owl III,

for letting me know when it was break time.

Abstract

The primary aim of this study was to develop a coherent basis for analysis of conditional nearest-neighbor spatial associations of wildlife distributions. The analysis involves the development and use of tessellation diagrams coupled with statistical measures of conditional spatial association. First, a new modification to the Quickhull algorithm is proposed for the creation of the convex hull. The convex hull is the geometric precursor to the tessellation diagram. Second, a new algorithm for the development of Delaunay and Voronoi tessellation diagrams is proposed. Third, a tessellation statistic for analyzing conditional locational interdependency is applied to a set of moose location information and the applicability of the method for wildlife analysis is reviewed.

Acknowledgments

I would first like to extend my thanks to my three supervisors, Prof. Jim Beck, Prof. Bob Hudson and Prof. John Hodgson. My two co-supervisors, Prof. Jim Beck and Prof. Bob Hudson have over the years helped me secure funding, given me critical advice on algorithm issues and supported this project with enthusiasm and encouragment, Thankyou Jim and Thankyou Bob, for your humour, support and advice. I would also like to extend my thanks to Prof. Hodgson, though he came on board in the latter thro's of this project, he has taken the time to review the manuscript and has given sound advice on its improvement.

I would like to extend my gratitude to Terry Osko. The unselfish act of sharing his many years of moose location data with me was without question instrumental in the success of this project. For his time and effort over the many years to coordinate and collect the hundreds of moose locations, for his advice and interest in my project and again for his kindness, I extend my sincere gratitude.

I required a great deal of input and support to complete this project and would like to extend many thanks to my Ministik colleagues, Jay Gedir, Shelley Pruss, Noble Donkor and Jason Galbraith, and the many others, for their support and advice over the years. It has been a great pleasure discussing the exciting wildlife projects in progress there. I wish each and every one of you the greatest of success in your careers and in your personal lives.

I would also like to thank the many people who over the years gave me help or advice, such as the office staff in renewable resources, the staff at grad studies, the staff in the SIS lab and the many others who have given up their time to help me. Thankyou.

I would like to thank Graduate Studies and Research for funding this project, as the funding was well needed. Thankyou.

Table of Contents

1. Introduction **1**

1.1 Bibliography 7

2. The Modified Quickhull Algorithm **9**

2.1 Introduction 9
2.2 Definition of Convex Hull 11
2.3 Overview of the Quickhull Algorithm 12
2.4 Quickhull Modifications 18
2.5 Algorithm Evaluations 23
2.6 Summary 26
2.7 Bibliography 28

3. Voronoi Tessellation **30**

3.1 Introduction 30
3.2 The Hull Triangulation Process 32
3.3 Pre-Insertion Processes 34
3.4 Edge Swapping Routines 36
3.5 Merging and Checking 39
3.6 Voronoi Tessellation 41
3.7 Summary 43
3.8 Bibliography 44

4. Statistical Tessellations of Moose Point Locations **45**

4.1 Introduction 45
4.2 Probability Theory 46
4.3 Mathematics and Geometry 49
4.4 Moose Analysis 57
4.5 Summary 60
4.6 Bibliography 62

5. Discussion and Conclusions **64**

5.1 Bibliography 68

List of Figures

Figure 2.2.1. A convex hull........ 11
Figure 2.2.2. A non-convex hull........ 11
Figure 2.3.1 The hull relative to the quadrilateral........ 12
Figure 2.3.2. The extreme quadrilateral........ 12
Figure 2.3.3. Quadrat assignments........ 13
Figure 2.3.4. The directed line test........ 14
Figure 2.3.5. An exterior Crossings test........ 15
Figure 2.3.6 An interior Crossings test........ 15
Figure 2.3.7. A negative Directed Line test........ 17
Figure 2.3.8. A positive Directed Line test........ 17
Figure 2.4.1. The Quarilateral Triangle Fan........ 18
Figure 2.4.2. Geometry of the Four edged model........ 19
Figure 2.4.3 Geometry of the Three edged model........ 19
Figure 2.4.4. Hull edge geometry........ 20
Figure 2.4.5. An exterior Triangle Fan test........ 21
Figure 2.4.6. An interior Triangle Fan test........ 21
Figure 2.4.7. The Triangle fan test........ 22
Figure 2.4.8 Hull path at time t+1........ 22
Figure 2.5.1. Quadrilateral point assessment graph........ 23
Figure 2.5.2. Processing times for Throw-away algorithms........ 24
Figure 2.5.3. Comparison of Original versus Modified Operations........ 25
Figure 3.2.1. The initial hull triangulation........ 32
Figure 3.3.1. The edge paving process. Dashed lines are those used in the directed line tests........ 34
Figure 3.3.2. Triangle splitting geometry........ 35
Figure 3.4.1. Delaunay circle properties........ 36
Figure 3.4.2. Radius test configuration........ 37
Figure 3.4.3. Pre-Swapped Triangles........ 37
Figure 3.4.4 Post-Swapped Triangles........ 38
Figure 3.5.1. The Delaunay tessellation........ 40
Figure 3.6.1. Tile triangulation........ 41
Figure 3.6.2. The Voronoi diagram (solid lines) and the Delaunay diagram (dashed lines) from the test data........ 42
Figure 4.2.1. The total probability sample space........ 47
Figure 4.3.1. Tile triangulation........ 51
Figure 4.3.2 Obtuse triangles........ 51
Figure 4.3.3. Disk in tile geometry........ 52
Figure 4.3.4 Disk area to be integrated........ 54
Figure 4.4.1. The May distribution of moose (black circles) and the Voronoi tiles centered on the lake centroids (gray squares). Note the boundary location used for the statistics (thick black lines)........ 59
Figure 4.4.2. Distribution of the probability of random variable R........ 59

List of Tables

Table 2.5.1. Average Cpu time in seconds for "Throw-away" algorithms. 25
Table 3.3.1. Insertion of initial point into the triangle segment records. 35
Table 3.4.1. Triangle segment record set.. 39
Table 4.4.1. Statistics for the conditional interdependency of moose with lake centroids. 58
Table 4.4.2. Disk values for Rc Calculations. 59

1. Introduction

The primary focus of wildlife distributional analysis is to determine if the animals distribution pattern is in some way affected by a specific causal agent, generally a landscape feature, an environmental gradient or a man-made disturbance. Inherent to this problem is the relationship between the environmental variables spatial arrangement and the affects the arrangement has on the animals distribution pattern. Even after decades of research there has been limited success in deriving a concrete unbiased statistical methodology that can provide insight into these distributional processes. It is the development of a spatial statistical application to answer questions like these that is the primary focus of this thesis.

Certainly, the problem has not been a lack of methodology, as there have been many statistical methods designed to measure either the dispersion or arrangements of wildlife point patterns. Unfortunately, all of the common dispersion techniques, like the nearest neighbor methods (see Pielou, 1969; Diggle, 1983 and Upton and Fingleton 1985) and the quadrat methods (see Getis, 1964 and Lee, 1974), which are used most often, suffer from biases brought about by an arbitrary determination of the study area boundary or through the use of statistics which depend upon the density of the observations. Thus we attempt to remove both the arbitrary boundary problem and the density dependency factors by using the more powerful arrangement techniques. Specifically those that utilize Voronoi tessellation diagrams (see Boots and Getis, 1988, 17-85 pp.).

Unraveling the history of the Voronoi tessellation diagram leads across numerous scientific disciplines and through many fascinating articles in several languages. The earliest known example of the use of tessellation geometry was from an astronomical diagram drawn by Descartes around 1629. A few centuries later the German mathematician Dirichlet (1850) published the first known mathematical treatise on the subject. However, it wasn't until subsequent mathematical foundations were developed by the French mathematician Voronoi (1908) that tessellation began to find a place in geometry. By the early 1920's German mathematicians like Niggli (1927) began to use the techniques to further research in the field of cellular networks. The next major contribution came from Delaunay (1934) who discovered the dual geometric form of the Voronoi, which consequently bears his name. Meanwhile, analytical works by the meteorologist Thiessen (1911) provided the first glimpses of the enormous analytical potential of the diagrams. However, the complex nature and tedious work involved in tessellation calculations led most tessellators of the mid 1900's to spend research time on finding better algorithms to improve diagram development as opposed to furthering the statistical frontiers. Certainly the advance of the computer was a turning point for the transition from diagram development to statistical analysis. The early statistics developed by researchers such as Neyman (1939), Robbins (1944), Garwood (1947) and Bronowski and Neyman (1945) laid the foundations for the tremendous strides in tessellation statistics by Boots (1974), Okabe and Miki (1984), Okabe *et al.* (1992), Okabe and Sadahiro (1994) and Okabe and Yoshikawa (1989). More recently, the topic of spatial statistics has gained popularity though it is still far from the mainstream, as are most spatial statistics of this century. For a thorough historical summary, consult Okabe *et al.* (1992).

To begin, tessellation in its formal sense is the replication of a figure or object. A group of tessellation objects fit together much like the interlocking pieces of a puzzle. This is the true form of tessellation. There is a great deal of information on the topic available across the internet. Voronoi tessellation and Delaunay tessellation are not true tessellations in the sense that the shapes or objects created are not identical, though they are interlocking. These tessellation diagrams are derived using locational information provided by the user. The information generally consist of points, lines or areas, describing the spatial location of landscape features, for example, well sites, lakes or forest cutblocks. A tessellation of point features, like the centroids of a group of lakes, would create a boundary around each centroid. The boundary equally divides the area between two adjacent centroids Therefore, the boundaries separate each centroids influencing area from all other centroids' respective areas. Each area can be thought of as a quadrat which provides a means to statistically analyze an observed set of locational information, like a series of x,y coordinates of moose locations with respect to the lakes.

The tessellation of point distributions has been extensively studied and there are many methods to choose from. Most methods begin by calculating the outer boundary of the feature points, formally called the convex hull. The rapid calculation of the hull is perhaps the most widely researched in computational geometry. One of the earliest known algorithms for its calculation, called the Quickhull, was developed by two McGill University researchers, Akl and Touissant (1978). Unlike most of its predecessors the Quickhull is a geometric sorting procedure as opposed to a mathematical one. Specifically, it sorts points using geometric primitives, like triangles, to determine the position of each point relative to an initial boundary.

Of the many hull forming algorithms now available for study, there exists only a handful of strategies. The two most common are the divide and conquer method and the incremental method (see Tsai, 1993 and Green and Sibson, 1977). The essence of the divide and conquer method is to divide a point dataset into two halves along the x axis, from each half form a separate convex hull and then conquer the problem by merging the two hulls together. In contrast, the incremental algorithm, which has many forms, essentially creates an initial triangle of three outer or inner points, then adds single points to the edge of this triangle, expanding it, until there are no longer any points to test. Both methods have had many pre and post processing improvements over the years and both are classified as deterministic algorithms, because they impose strict ordering to the selection and processing of point information. The original Quickhull algorithm is a mixture of these two methods.

In more recent years, there has been a trend toward the use of Las Vegas style algorithms for convex hull development (Clarkson *et al.* 1989). These algorithms select points based on a randomized selection process. They have an expected running time that matches the lower time constraint required to solve the convex hull problem (see Boas, 1980 and Avis, 1980). Since we are interested in redesigning the core equations in the original Quickhull algorithm, we have added in a randomization process but not as the primary modification, hence we have chosen to ignore true algorithm randomization as it can easily be added later.

In my search for a linear time algorithm, with which to construct the hull, I propose the theory that the sequential ordering of geometric primitives would provide maximum topological efficiency, under both stable and random spatial change, which in turn would maximize the efficiency of finding those points which make up the hull. To maximize topological efficiency means to assign a series of non-overlapping geometric objects, such as triangles, to completely fill an irregular open space, like a polygon, in such a way that the objects used is minimized. Once found, the minimum number of triangles should provide an optimum solution for determining if a given test point falls inside the polygon. An example of the use of this theory is the application of the sorted Triangle Fan algorithm to solve a point in polygon problem (see Haines, 1994). This theory also extends to the development of the Delaunay diagram.

The second part of diagram development consists of identifying the nearest neighbors of every feature point. A common strategy for solving this problem is to use the convex hull to determine the position of a point relative to its' neighbors. Once the two neighbors are located a closed line segment is drawn which connects the point to each of its two nearest neighbors, thereby forming a triangle. The resulting triangulated set is known as a Delaunay tessellation diagram.

As previously inferred, the triangular topology of the Delaunay lends itself to the creation of the Voronoi, which is sometimes referred to as the dual graph. This utility of this duality was first reported by Boots in 1974 and since then it has become a common practice to create one diagram from the other. The creation of the Voronoi from the Delaunay is by simply attaching together the perpendicular bisectors of each line in the Delaunay's triangulation (see Aurenhammer, 1991). The resulting linework forms individual regions called Voronoi tiles. These regions represent the area of influence around a given feature point. The tile boundaries thus represent a division of influence between two adjacent features.

I proposed to develop a Delaunay diagram from a convex hull, by using the Triangle Fan algorithm which is a typical geometric example of the divide and conquer strategy. The topology of the Triangle Fan linework is similar to that used in Delaunay diagrams and therefore provides a paralleled approach to diagram creation. Though it does not have the point-in polygon speed of the gridding algorithms, the Triangle Fan test does not have the memory overhead nor the secondary setup time inherent to the gridding strategies (Haines, 1994). It is therefore very simple to implement as a computer program.

Unlike the convex hull and tessellation geometry, tessellation statistics have received little attention from the research community. A handful of Japanese researchers, headed by Professor A. Okabe from the University of Tokyo, began the statistical development in the early 1980s. The first examples were presented by Okabe and Miki in 1984. They combined conditional probability theory, integral calculus and tessellation geometry to determine the effects subway stations had on the distributions of local retail stores. The results of this study opened the door for others to explore the effects of roadways and parks on the distributions of high cost apartments. Unfortunately, the complexity of the procedure has somewhat stunted its' growth to this point. A complete review of the methodology is provided herein, however a reader is advised to consult the original research paper by Okabe and Miki (1984).

Any attempt at spatially analyzing wildlife distributions must be critically linked to the behavior of individual animals. This is one of the key features of the tessellation statistic. It is highly sensitive to the spatial variation in distribution data brought about by differences among individual activity patterns. This sensitivity lends itself to the analysis of any landscape feature or point distribution thought to influence the behavior of the study animals. An example would be the influence of water bodies on the distribution of moose.

With this aside we open the thesis with a chapter devoted to the construction of the convex hull detailing our proposed modifications to the well known Quickhull algorithm. In the second chapter we discuss the formation of both the Delaunay and Voronoi tessellation diagrams, outlining the basic methods used and describing our technique in detail. In the third chapter we review the basis for the tessellation statistics, apply the techniques to a wildlife problem and summarize our findings. In the final summary I recap the projects' overall aim, discuss the success of our proposed diagram modifications and further discuss the use of Voronoi tessellation as a statistical tool for analyzing wildlife information.

1.1 Bibliography

Akl, S.G. and G.T. Toussaint. 1978. A fast convex hull algorithm. Inform. Proc. Lett. 7(5): 219-222.

Aurenhammer, F. 1991. Voronoi diagrams - A survey of a fundamental geometric data structure. ACM Comp. Surveys. 23(3): 346-405.

Avis, D. 1980. Comments on a lower bound for convex hull determination. Inform. Proc. Lett. 11(3): 126.

Boas, P.E. 1980. On the Ω (n log n) lower bound for convex hull and maximal vector determination. Infor. Proc. Lett. 10(3): 132-136.

Boots, B.N. 1974. Delaunay triangles: An alternative approach to point pattern analysis. Proc. Assoc. Amer. Geog. 6: 26-29.

Boots, B.N. and A. Getis. 1988. Point Pattern Analysis. Sage Publ. Inc. London, England.

Bronowski, J. and J. Neyman. 1945. The variance of the measure of a two dimensional random set. Ann. Math. Statist. 16: 330-341.

Brown, K.Q. 1979. Voronoi diagrams from convex hulls. Infor. Proc. Lett. 9(5): 223-228.

Clarkson, K.L., Tarjan, R.E. and C.J. VanWyk. 1989. A fast Las Vegas algorithm for triangulating a simple polygon. Discrete Comp. Geom. 4: 423-432.

Delaunay, B.N. 1934. Sur la sphere vide. Bull. Acad. Sci. U.S.S.R. : Class. Sci. Math. Naturelle 7(6): 793-800.

Diggle, P.J. 1983. Statistical analysis of spatial point patterns. Academic Press, New York.

Dirichlet, G.L. 1850. Uber die reduction der positeven quadratischen formen mit drei unbestimmtem ganzen zahlen. J. Reine and Angew. Mathe. 40: 209-227.

Garwood, F. 1947. The variance of overlap of geometrical figures with reference to a bombing problem. Biometrika 34: 1-17.

Getis, A. 1964. Temporal land-use pattern analysis with the use of nearest neighbor and quadrat methods. Ann. Assoc. Amer. Geogr. 54: 391-399.

Green, P.J. and R. Sibson. 1977. Computing Dirichlet tessellations in the plane. Comp. Jour. 21(2): 168-173.

Haines, E. 1994. Point in polygon strategies. In *Graphics Gems IV*, P.S. Heckbert eds., Academic Press Inc., Cambridge., pp. 24-46.

Lee, Y. 1974. An analysis of the spatial mobility of urban activities in downtown Denver. Ann. Reg. Sci. 8: 95-108.

Neyman, J. 1939. On a new class of "contagious" distributions, applicable in entomology and bacteriology. Ann. Math. Statist. 10: 35-57.

Niggli, R. 1927. Die topologische stukturanalyse. Z. Kristallographie 65: 391-415.

Okabe, A. and F. Miki. 1984. A conditional nearest neighbor spatial-association measure for the analysis of conditional locational interdependence. Env. Plan. A. 16: 163-171.

Okabe, A. and T. Yoshikawa. 1989. The multi-nearest neighbor distance method for analyzing the compound effect of infrastructural elements on the distribution of activity points. Geog. Anal. 21(3): 216-235.

Okabe, A. and Y. Sadahiro. 1994. A statistical method for analyzing the spatial relationship between the distribution of activity points and the distribution of activity continuously distributed over a region. Geog. Anal. 26(2):152-167.

Okabe, A., B. Boots and K. Sugihara. 1992. Spatial tessellations : concepts and applications of Voronoi diagrams. John Wiley and Sons Inc. NewYork, N.Y.

Robbins, H.E. 1944. On the measure of a random set. Ann. Math. Statist. 15: 70-74.

Thiessen, A.H. 1911. Precipitation averages for large areas. Monthly Weather Rev. 39: 1082-1084.

Tsai, V.J.D. 1993. Fast topological construction of Delaunay triangulation's and Voronoi diagrams. Comp. & Geosci. 19(10): 1463-1474.

Upton, G. and B. Fingleton. 1985. Spatial data analysis by example. Volume I: Point pattern and quantitative data. John Wiley, Chichester, England.

Voronoi, G.M. 1908. Novelles applications des parameters continus a la theorie des formes quadratiques. J. Reine Agnew. Mathe. 134: 198-287.

2. The Modified Quickhull Algorithm

2.1 Introduction

Since its development in 1978, the Quickhull algorithm has undergone few changes. The model is generally very fast (O'Rourke, 1995) and for this reason alone improvements to the model are difficult. Recently the model was converted into a randomized Las Vegas style algorithm by Wenger (1997). Wengers modifications provided an O(n log h) model, which is independent of the input data distribution. Chan *et al.* (1995) also modified the Quickhull to provide an output-sensitive model. The original model is deterministic and has an expected runtime of O(n), but it is not clear whether it is distribution independent. In my review of the literature I found that most deterministic algorithms were not distribution independent. However, the general focus is on the reduction in overall processing time for random data sets and not distribution independence. In this study of convex hulls I propose to modify the Quickhull algorithm in an attempt to decrease the processing time for data points in R^2. Earlier investigations which compared many of the primary point-in-polygon algorithms, suggests that the Quickhulls' core tests are likely inferior to a variety of newer algorithms (Haines 1994). Whether an alternative, as suggested by Haines, could be incorporated into the Quickhull model was speculative.

The Quickhull's inherent use of geometric shapes to filter out the probable hull points provides a simple mechanism to test the differences between the various proposed alternatives. With respect to the Quickhull's divide and conquer strategy, there are two distinct filtering processes. The first process consists of filtering the initial dataset into points located in the interior of a derived quadrilateral with those that are exterior. This filtering process is intended to allow a faster testing of potential hull points than could be accomplished without one. Under most conditions the filtering does improve the actual selection of the hull points. However, this is not true for all algorithms as many of these techniques are quite complex, requiring reading and re-reading of subsets of the data to derive the pre-ordered state. Obviously, the key to a fast algorithm lies in its ability to order the data. The approach taken by almost all the convex hull algorithms reviewed herein excelerate the creation of a convex hull using some form of preliminary data ordering (Tsai 1993, Ohya *et al.* 1984, Brassel and Reif 1979, Dehne and Klein 1997, Maus 1984, Graham 1972, Preparata and Hong 1977).

The second process, the conquering of the problem, involves using these exterior points to find the convex path for each of the four edges of the derived quadrilateral. We apply the theory of geometric sequencing to devise a means to rapidly locate points as the hull path is formed. A quadrilateral edge, along with the section of hull that connects to it, can be considered as an irregular polygon, whose shape changes over the course of hull formation. The vertices of the hull edge form a natural triangular fan with the quadrilaterals edge, thus I chose to use triangles as the mechanism to divide up these polygons, which should maximize the polygons topological efficiency. In this way, the Triangle Fan algorithm is a natural choice as a modification to the original model.

Customarily, the divide and conquer operations are separated, both in the code and in the design of the model, however, this is not the case for the Quickhull. The flow of individual data points is sequential through the processes, allowing the complete sorting of a point, from start to finish, thereby eliminating a second handling of the data. Although this was not the method described in the original paper, it was proposed toward the end and I feel it is the superior method of all those originally proposed. Although, I have conveniently divided them out in the following text.

In the first section I give a formal definition of a convex hull. In the second section I review the original Quickhull model as described by Akl and Toussaint (1978). The third section provides descriptions of the proposed modifications and example calculations for all geometric primitives and the new algorithms. In the fourth section I provide an analysis of the original hull to those proposed. Lastly, I summarize my findings.

2.2 Definition of a Convex Hull

The formal definition of a convex hull is described as the set of points which form a convex boundary around all data points in the sample space. This implies that all hull points are connected in a closed loop, called a polygon and that the interior angle at any given hull point is less than 180^0 (Figure 2.2.1). A hull point with an angle that exceeds 180^0 would create a dent in the boundary thus rendering it non-convex (Figure 2.2.2). This does not mean that the sample points are no longer encompassed, in fact they must be otherwise it is not a hull. Another less obvious constraint is that the line segments connecting the hull points do not overlap. This special case of an inverted polygon, though convex by the above definition, is actually not convex due to its overlapping line segments.

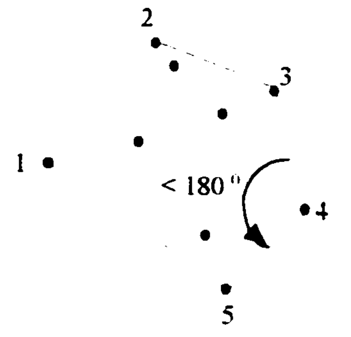


Figure 2.2.1. A convex hull

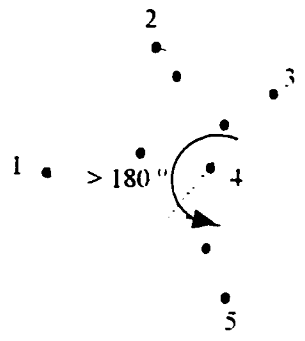


Figure 2.2.2. A non-convex hull

This and other degenerate cases are reviewed by Schorn and Fischer (1994). Convexity is an issue in that the properties and utility of the polygon change drastically once it becomes non-convex, and this can render some tessellation forming algorithms unusable. A list of formal convex hull definitions is found in O'Rourke (1995, pp. 71-72).

2.3 Overview of the Quickhull Algorithm

Akl and Toussaint (1978), recognized that a convex hull could be formed from any three initial geometric shapes; a quadrilateral, a triangle or a line. As mentioned in the chapter introduction, their general strategy is essentially a geometric form of the divide and conquer principle. To describe the strategy using the quadrilateral, the algorithm begins by determining the four extremal points in the dataset. These four points become the four corners of the extreme quadrilateral (Figure 2.3.1). Note that these four extreme points are all on the hull (Figure 2.3.2). It then proceeds to eliminate points that fall in the interior of the quadrilateral, as these points are not on the hull. Akl and Toussaint called this the "throw-away principle". The throw away principle uses Algorithm 112 as cataloged by the ACM (Shimrat, 1962). The original equation has an error in it, but fortunately this was corrected by Hacker (1962). This algorithm is commonly called the "Crossings test" (Haines 1994). A mathematical example of Shimrats algorithm, including the geometry, is given in Example 2.3.1. at the end of this section.

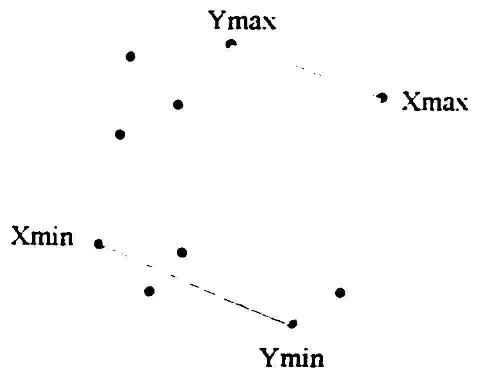


Figure 2.3.1. The extreme quadrilateral.

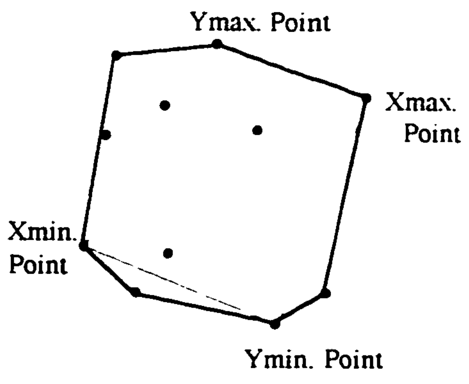


Figure 2.3.2. The hull relative to the quadrilateral.

The second stage in the process is to divide up the remaining exterior points into four quadrat groups according to the region that they fall into (Figure 2.3.3). For efficiency, the division of the points occurs during the "throw-away" testing. These quadrat points are then sorted by their x coordinates in the following way; ascending order for points in quadrats 1 and 2 and descending for points in quadrats 3 and 4. The quadrilateral used in this manner correlates to the "divide" part of the divide and conquer method. To conquer the problem all that remains is to develop the convex path for each quadrat.

Obviously, each path begins at one extreme corner of the quadrilateral and finishes at the next. The focus is to eliminate all quadrat points not on the path. Those left in the quadrat list have already been properly sorted, hence they form a proper convex path for the given quadrat. To create this convex path, Akl and Toussaint developed an algorithm, called the "directed line test". This test is well recognized as the core test for many forms of the incremental method. It is performed by calculating the area of a triangle and noting the sign of the resulting value (O'Rourke 1995). The first directed line test includes one corner of the quadrat edge, call it a, the first sorted point in the quadrat list, call it c and the next point in the quadrat list, call it b. If the area, as calculated from a to b to the test point c, is a positive value, point c is to the left of the directed line ab. If the value is negative, point c would be to the right of the line ab (Figure 2.3.4). In the quadrat cases, negative values are inner to the existing hull path, so are removed from the list. This iterative processing is repeated until all negative points are removed from each quadrat list. The directed line test is often called

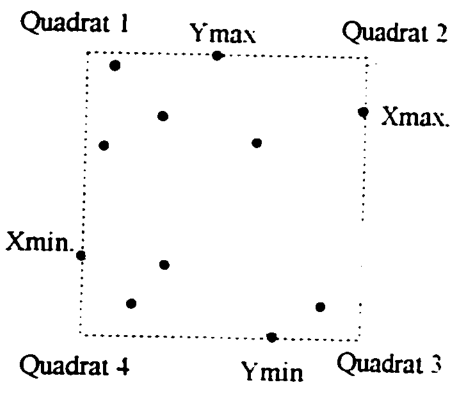


Figure 2.3.3. Quadrat assignments.

the Clock test because positive values are formed by a counter-clockwise (CCW) orientation of the points and negative values result from a clockwise (CW) orientation. A mathematical example of the test, including the geometry, is provided in the example 2.3.2, at the end of this section.

The original paper on the Quickhull also suggests that the processing of points can be done by repeatedly using the throw-away principle and dropping the use of the sorting all together. Strictly speaking the algorithm does not require the sorting mechanism, as the traversal of the hull, using the defined directed line test, can determine the hull points. The sort was added to speed up the selection of points to test. The sorting accomplished a faster termination of the hull traversal routine, however, the direct traversal method is faster, since at that time it is likely the fastest sorting method available had an expected run time of only O(n log n). As outlined in the paper, eliminating the sort will achieve an expected runtime of O(n). A complete review of the use of Big O notation for algorithm classification is provided in an excellent article by Schneier (1994) on NP-completeness. This completes the overview of the basic structure and function of the original Quickhull algorithm. The modifications which follow are based on an evaluation of the effectiveness of the standard quadrilateral form.

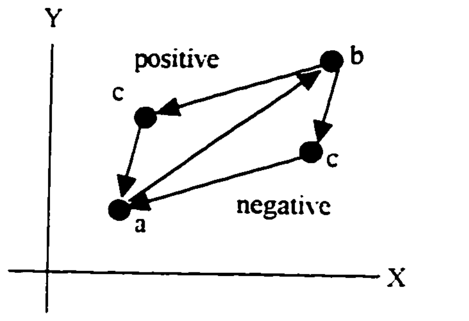


Figure 2.3.4. The directed line test.

Example 2.3.1 Shimrats Crossings Test (Shimrat 1962).

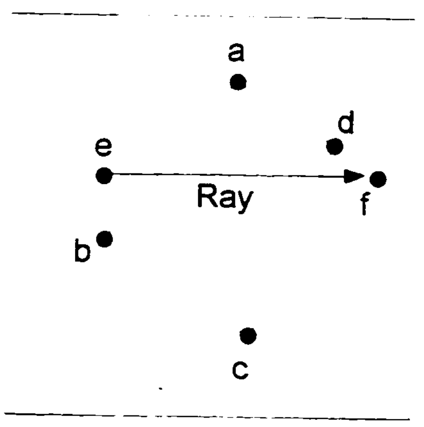


Figure 2.3.5. An exterior Crossings test

If points a to f have coordinates as follows:

b (2.4). c (4.2). d (6.6). a (3.5. 7)
e (1.8. 5). f (7.5).

with the test point being e and a test ray shot to point f. then the crossings test is calculated as follows:

First check all endpoints to determine if they cross the ray by comparing them to e.y.

if (a.y > e.y) and (b.y < e.y). then I know that edge ab and edge cd have endpoints on either side of the ray but I do not know if either edge actually crosses the ray. so I must solve for the x value of the intersection using the Crossings formula. If the inequality is true.

(b.x - ((b.y - e.y)(a.x-b.x) / (a.y-b.y))) >= e.x.
(2 - ((4 - 5)(3.5-2)/(7-4))) >= 1.8.
(2 - ((1.5)/(3))) >= 1.8
2.5 >= 1.8

then the ray crosses the edge.

There are many variations to the testing procedure which fit particular circumstances. Here keeping

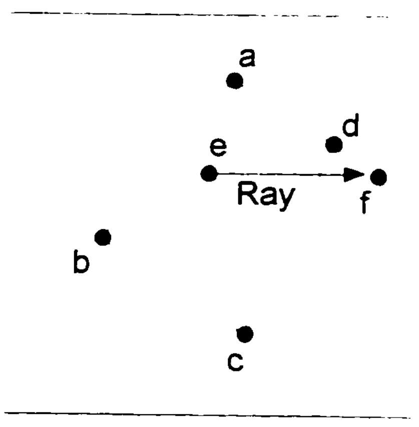


Figure 2.3.6. An interior Crossings test

If points a to f have coordinates as follows:

b (2.4). c (4.2). d (6.6). a (3.5. 7)
e (3. 5). f (7.5). g (1.8. 5)

with the test point being e and a test ray shot to point f. then the crossings test is calculated as follows:

Again check all endpoints to determine which edges have endpoints on either side of the ray. the edge cd is a candidate so is edge ab. so I calculate as before.

(b.x - ((b.y - e.y)(a.x-b.x) / (a.y-b.y))) >= e.x.
(2 - ((4 - 5)(3.5-2)/(7-4))) >= 3
(2 - ((1.5)/(3))) >= 3
2.5 >= 3

which is false. Calculating the second edge cd I find.

(d.x - ((d.y - e.y)(c.x-d.x) / (c.y-d.y))) >= e.x.
(6 - ((6 - 5)(4 - 6) / (2 - 6))) >= 3
(6 - ((-2) / (-4))) >= 3
5.5 >= 3

track of the edges being tested works best. Given that the initial edge comparison tests all four edges. I find it takes a minimum of nine mathematical operations to complete the task.

which is true. so I conclude that point c is interior to the quadrilateral. The total number of tests is at least twenty two.

Example 2.3.2 Akl and Toussaints Directed Line Test (Akl and Toussaint 1978)

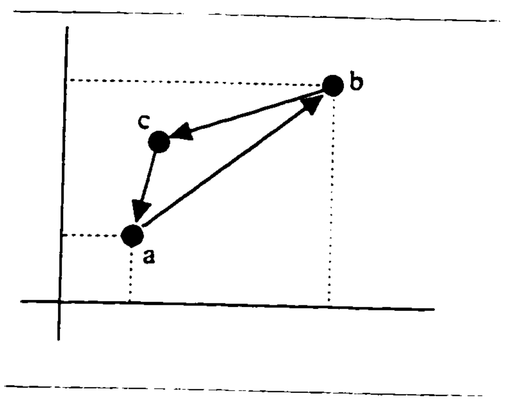


Figure 2.3.7. A positive directed line test.

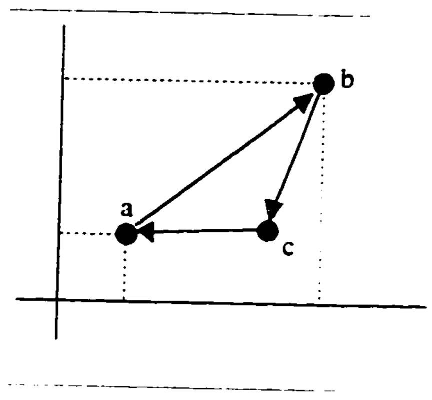


Figure 2.3.8. A negative directed line test.

If the points a to c have the following coordinates:

a(2.2), b(4.6) . c(2.2. 4) and d(3. 2.2).

then the directed line test from a to b with c as the test point is calculated as.

s = (b.x - a.x)(c.y - a.y) - (b.y - a.y)(c.x - a.x)
s = (4 - 2)(4 - 2) - (6 - 2)(2.2 - 2)
s = 2 (2) - 4 (0.2)
s = 3.2

Since the result is positive I conclude that point c is to the left of the directed line ab. To confirm the test correctly assigns the sign, a negative directed test using the same directed line ab with d follows.

If the points a to c have the following coordinates:

a(2.2), b(4, 6) and c(3, 2.2)

then the directed line test from a to b with c as the test point is calculated as.

s = (b.x - a.x)(c.y - a.y) - (b.y - a.y)(c.x - a.x)
s = (4 - 2)(2.2 - 2) - (6 - 2)(3 - 2)
s = 2 (0.2) - 4 (1)
s = -3.6

Overall there is a total of seven mathematical operations per line test, five subtractions and two multiplication's. The result is negative. This verifies the signs of the test results. Again the same number of operations per directed test.

Another important use of this test is for the area of a triangle. The calculated value is twice the area of the triangle abc, thus taking the absolute value and dividing in half gives the true area.

2.4 Quickhull Modifications

The first modification was to replace Shimrats' "Crossings" algorithm, which was used for the throw-away principle, with Greens'(1993) Triangle Fan algorithm. The modification divides the extreme quadrilateral into two triangular regions, using a single shared edge. This is done by joining the Ymax point to the Ymin point or secondly by joining the Xmax point with Xmin point. Doing this creates two triangles, one with corners Ymax:Ymin:Xmin and the other with corners Ymax:Xmax:Ymin. A single directed line test using the triangles' shared edge, call it line Y:Y, places a test point in either triangle one or two (Figure 2.4.1). A second and possibly third test, using the exterior edges of the point's triangle, confirms the point's location with respect to the quadrilateral's edges. A second test will determine if the test point is exterior to the quadrilateral or that a third test will be needed. A third directed line test will determine the points position as being either exterior or interior to the quadrilateral.

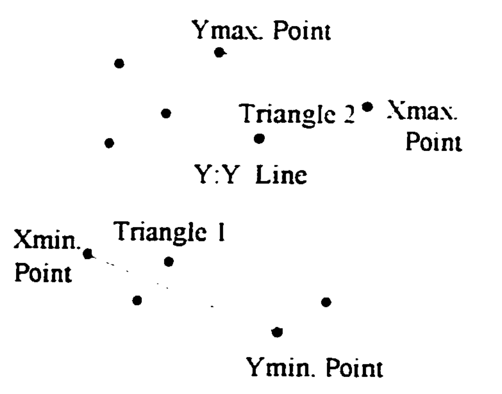


Figure 2.4.1. The Quadrilateral Triangle Fan

Two other throw-away methods were also evaluated. A three edged directed line test which uses the "edge-paving" technique and a four edged directed line test which simply tests every edge of the quadrilateral in order. The three-edged model balances the testing of points regardless of the data distribution. It does this by dividing the quadrilateral into four quadrats (Figure 2.4.2). Every point is tested exactly three times; once using line Y:Y, once using line X:X and lastly by the appropriate quadrilateral or quadrat edge. This preserves the essence of the algorithms geometric divide and conquer principle. In contrast, the four edged model attempts to average out the number of edges tested by testing all four edges in sequence, using the directed line test. Should a point be found exterior to a given edge the test is completed, therefore it tests two edges on average for datasets evenly distributed across the plane (Figure 2.4.3). An example of the geometry and mathematics of the both the Triangle Fan test and edge paving test is provided at the end of this section.

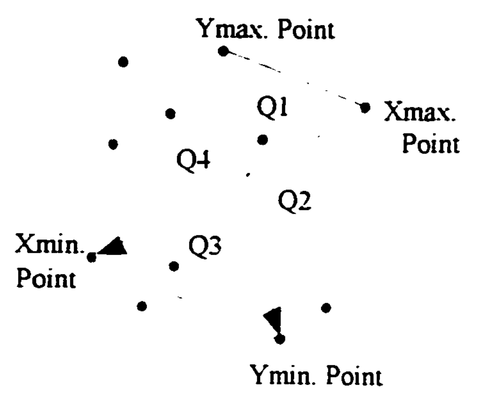


Figure 2.4.2 Geometry of the Three edged model.

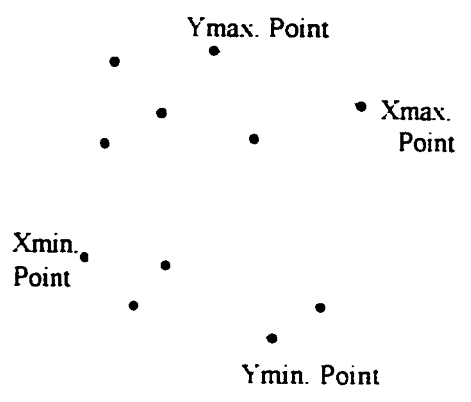


Figure 2.4.3. Geometry of the Four edged model.

The second modification to the original algorithm, removes step two, the point sorting. It also replaces the iterative processing of the points along the convex path with a hybrid triangle fan test (see Badouel 1990, Green 1993). The fan test used follows the test proposed by Green, with slight modifications. To execute the Triangle fan the first data point found exterior to the initial extreme quadrilateral is attached to the quadrat corners and thus forms an initial convex path, whose shape is a triangle. Quadrat points are then tested to determine if they fall inside this triangle. If they do they are eliminated from further consideration. If the point is outside, a second triangle is formed using one quadrat point and the two test points. As arranged, the first triangle has the longest edge and the next adjacent triangle to it has the next longest edge and so forth as the fan grows. This provides a natural sorting mechanism for each triangle on the fan (Figure 2.4.4). Recognizing that each triangle in the fan shares an edge with either the quadrat edge or an adjacent triangle, a test for inclusion would only require the testing of two edges per triangle. Moreover, a quicker determination of the point with respect to the hull, is found by knowing which edge of the triangle the point is exterior to. If the test point is exterior to the triangles "hull" edge, as designated in figure 2.4.4, the test is complete, otherwise the next adjacent triangle must be tested. Processing a point via the fan test completes the initial stage of path development.

To complete the convex path every point found outside a hull edge must undergo a traversal of the existing edge to properly incorporate the point into the path array. The traversal is done to remove any existing hull points which would fall interior to the convex path once the new point was added. The check for tangency uses the directed line test and is therefore the same operation as performed in the original algorithm, only very selectively. In summary, the addition of each new hull point requires both finding the proper location in the hull to insert the point, as well as the elimination of any existing hull points that are now interior to the newly formed convex path. The construction of the hull edge is therefore very dynamic, as points are added others are deleted until the final path is formed.

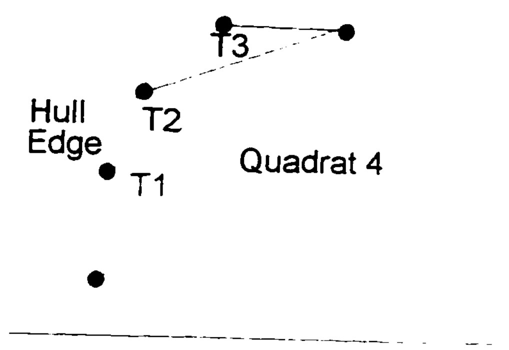


Figure 2.4.4. Hull edge geometry

In the next section we present the results of a comparison between the original Quickhull processes, the modified Quickhull, and the proposed quadrilateral variations. The comparisons were accomplished by programming each algorithm as inline code and incorporating them into the exact same C++ method (function). Thus the processes were almost identical except for the slight variations in control structures.

Example 2.4.1 The Quadrilateral Triangle Fan Test

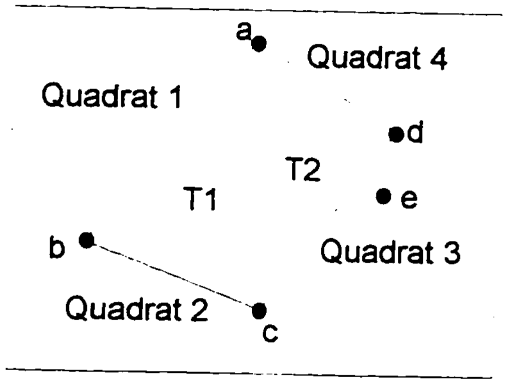


Figure 2.4.5. An exterior Triangle Fan test

The two triangles T1 and T2 share a common edge represented here by the dashed line. We can readily determine if the test point e is on the left or right of the this common edge by a simple directed line test using line ac. A second directed test using either quadrat edge, for the given T, could determine if the point is exterior, giving a fast two edge test. A common modification to increase the overall speed of the test is to test the longest edges first. Given that the edges are not sorted there is a fifty percent chance of selecting the correct edge first. Given that points a to e have coordinates as follows:

a(3.5, 7), b(2, 3), c(5,2), d(6, 6), e(7, 5).

We calculate e's position by the following.

s = (c.x - a.x)(e.y - a.y) - (c.y - a.y)(e.x - a.x)
s = (5 - 3.5)(5 - 7) - (2 - 7)(7 - 3.5)
s = 14.5.

so I know e is to the left of the directed line ac, next I arbitrarily select line cd to test.

s = (d.x - c.x)(e.y - c.y) - (d.y - c.y)(e.x - c.x)
s = (6 - 5)(5 - 2) - (6 - 2)(7 - 5)
s = -5.

so I have confirmed e is exterior to the line cd and the test is done.

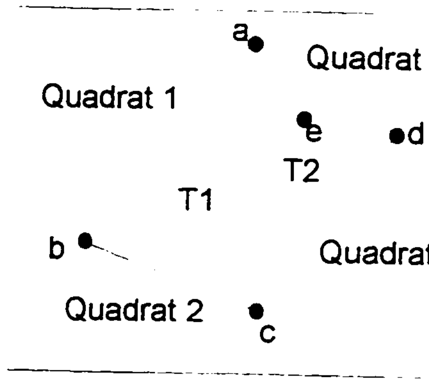


Figure 2.4.6. An interior Triangle Fan test

If the points have the same coordinates except the test point I have:

a(3.5, 7), b(2, 3), c(5,2), d(6, 6), e(5.5, 5).

We calculate e's position by the following.

s = (c.x - a.x)(e.y - a.y) - (c.y - a.y)(e.x - a.x)
s = (5 - 3.5)(5 - 7) - (2 - 7)(5.5 - 3.5)
s = 7.

so I know e is to the left of the directed line ac, next I arbitrarily select line cd to test.

s = (d.x - c.x)(e.y - c.y) - (d.y - c.y)(e.x - c.x)
s = (6 - 5)(5 - 2) - (6 - 2)(5.5 - 5)
s = 1.

so I have confirmed e is interior to the line cd lastly I test directed line da.

s = (a.x - d.x)(e.y - d.y) - (a.y - d.y)(e.x - d.x)
s = (3.5 - 6)(5 - 6) - (7 - 6)(5.5 - 6)
s = 3.

confirming the point is interior to the triangle T2.

Example 2.4.2 The Hull Triangle Fan Test

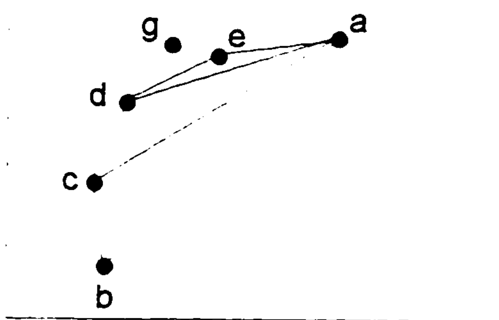


Figure 2.4.7. The Triangle fan test

If line ab represents one edge of the quadrilateral and a to e. make up the hull points at time t then to test point g for inclusion to the hull requires.

a directed line test using line ab.

which determines that point g is exterior to the inner quadrilateral. next I test the remaining inner triangle edges in sequence as follows.

a directed line test using line ac.
a directed line test using line ad.
a directed line test using line ae

From these tests the position for insertion of g. ie after point e. is known. but the remaining hull edges must be traversed to eliminate points like e which are non-convex once g is added.

The final traversal begins with a directed line test from e to d which tags point e for deletion. because g is twice left of e. The rest of the exterior edges are tested in the same manner. The hull at time t+1 now looks like Figure 2.4.7.

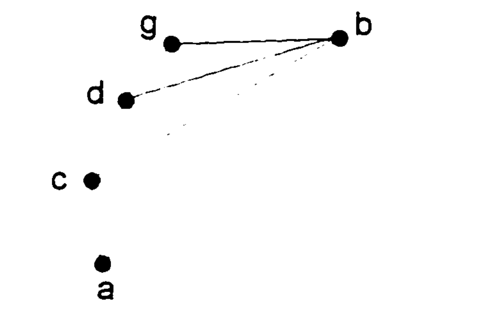


Figure 2.4.8. Hull path at time t+1

An evaluation of the these operations quickly reveals that a direct traversal of the edge for every test point would result in a greater number of directed line tests. This is due to the fact that the triangles can terminate the testing should the point fall inside an inner edge and this in turn results in fewer overall tests. Moreover. the triangular topology is an excellent example of maximizing topological efficiency. Notice the inner triangle edges sum to one less than the edge of the hull and that the interior edges are naturally sorted by length. beginning with the longest edge first.

2.5 Algorithm Evaluations

The original research paper on the Quickhull states that approximately one-half of randomly distributed data points would be interior to the extreme quadrilateral. To ensure the execution of the original algorithm is as claimed I randomly generated twenty-five points two hundred fifty times and measured the distribution of points with respect to the extreme quadrilateral. The original claim is true, approximately one-half of the points are eliminated by the use of the Crossings algorithm (Figure 2.5.1). With this information I began to search for a better alternative.

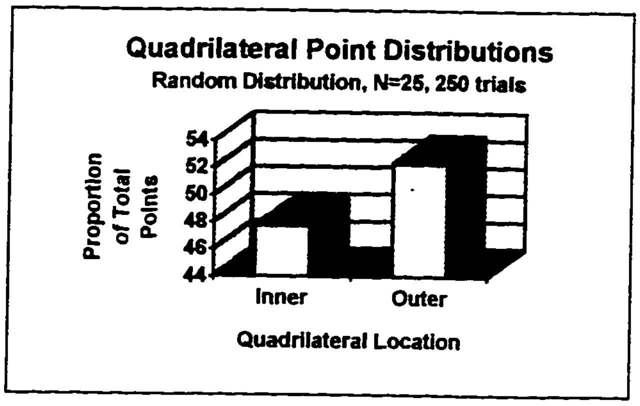


Figure 2.5.1. Quadrilateral point assessment

A first thought was to evaluate the Crossing test. The Crossings test depends upon the dataset's point distribution and more importantly upon the orientation of the initial quadrilateral. An improvement to this test would require the use of a triangle as the extreme geometric shape, however this would lead to a reduction in the number of initial points "thrown away" and an increase in the number of points processed via the hull filter, which could only lead to an increase in the processing time, since the hull filtering process is more complex. A review of "point in polygon" algorithms by Haines (1994) confirms the drawbacks of the Crossings test and suggests several others as replacements. The fastest replacement test for polygons with many sides is the Grid test, as it provides nearly constant time, but the Grid test has the unfortunate drawback of being very memory intensive to set up and manage. An even faster test for polygons with few sides is the Sorted Half-Plane (or Triangle Fan) test.

To ensure that my selected Triangle Fan replacement was indeed faster than the Crossings or the Three and Four Edge directed line tests, all algorithms were programmed in C++, using the same class and same hull path processing method. Trials were done in sets of twenty-five repetitions. Each repetition generated a randomly distributed point set of a given size. Five sizes were chosen going from one-thousand points in a set to five thousand points per set. The outcome of the twenty five trials was averaged for each of the four test algorithms. The results are shown in Figure 2.5.2. Our regression analysis with additional trials reveals that every test procedure was an O(n) algorithm (Table 2.5.1), with the Triangle Fan averaging the fastest computational time of those tested.

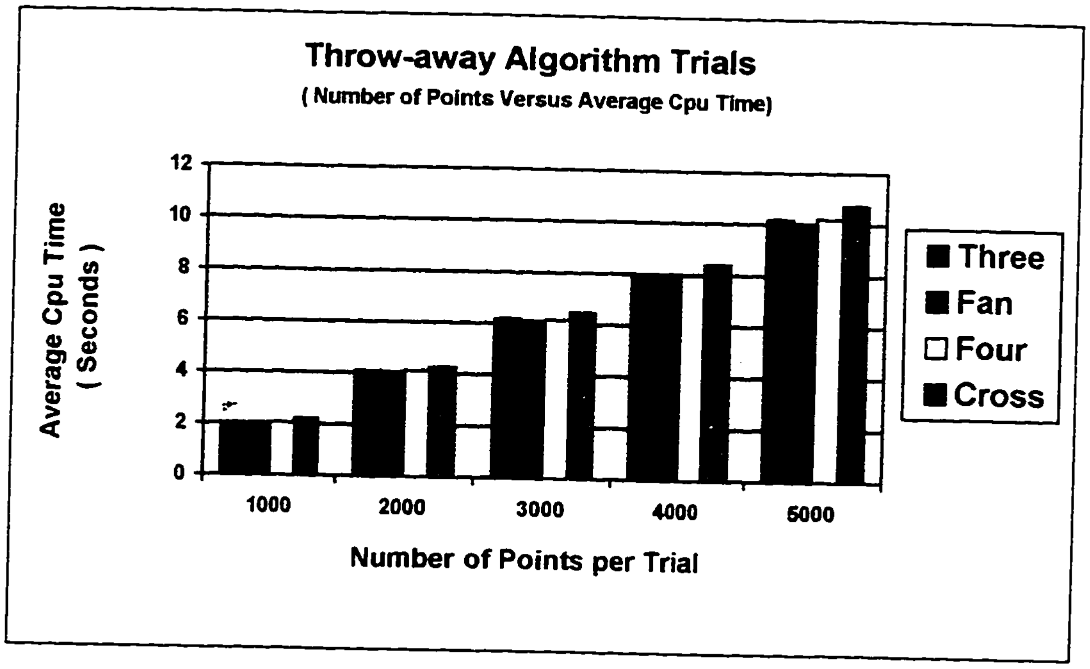


Figure 2.5.2. Processing times for Throw-away algorithms

The results are really not that surprising as the Triangle Fan test will have fewer operations, with respect to all other tests, on all datasets except those whose points consistently require a "third edge" assessment. The exception to this fact appears in the trials of four-thousand points. Aside from this minor variation the Triangle Fan still processes single points faster than any of the other proposed algorithms.

For evaluation purposes, the conquer portion or second modification of the algorithm which traverses the hull edges was divided into three separate hull operations; failed left tests, point deletions and point additions. The failed left tests are the sum of all directed

N	HULL PTS	THREE	FAN	FOUR	CROSS
1000	16.88	2.0216	2.024	2.048	2.1816
2000	20.6	4.0736	4.0292	4.108	4.2484
3000	21.16	6.1632	6.0948	6.1408	6.4464
4000	22.24	7.9192	7.9832	7.9552	8.3816
5000	22.12	10.1848	10.0336	10.2028	10.72
10000	24	19.4772	19.1364	19.3264	20.2184
50000	29.56	100.526	97.8628	98.6744	102.7412
Avg/Point	22.37	0.00200	0.00196	0.00198	0.00207
R^2	0.8901781	0.9999719	0.9999783	0.999978	0.9999797

Table 2.5.1. Average Cpu time in seconds for "Throw-away" algorithms.

line tests used to process points which did not result in the direct deletion or addition of the point to the hull. If a point is on the hull path but its' position is no longer convex due to the addition of another point, they are classified as deletes. The add operations are due to additions to the hull regardless of whether they are deleted later as the true path unfolds (Figure 2.5.3).

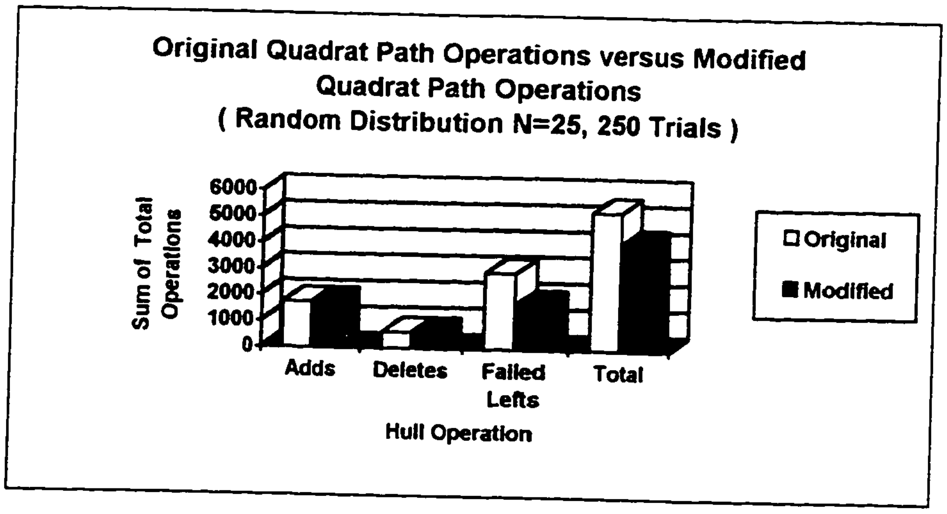


Figure 2.5.3. Comparison of Original versus Modified Operations

Figure 2.5.3 represents the total sum of each operation across 250 trials. By this figure the original Quickhull model spends most of its time resolving failed left tests. The Triangle Fan has practically the same number of additions and deletions, but fewer failed lefts, significantly improving the models overall performance.

2.6 Summary

In Akl and Toussaints words " the simplicity and speed of the proposed algorithm make it worth reporting". The Quickhull algorithm is indeed a very fast convex hull algorithm. Of the three proposed modifications to the quadrilateral point filtering process (throw-away process) none were non-linear. Furthermore, any one of the three proposed algorithms was, by my evaluation, better than the original model. The Crossing test algorithm itself is a very fast test, however the initial point evaluations to determine if the edge endpoints are on either side of the test point ray, is very costly. Even the optimization employed to reduce all unnecessary control structures could not provide a better result than those obtained by the others. Indeed, from the computer programming perspective the fastest algorithms are generally those having the fewest and least complicated control structures.

A review of the operational elements of the convex path construction reveals that the traversal of the hull edge is the most costly of the three path operations. The overhead incurred trying to determine a points location with respect to the hull edge leads to many failed directed line tests. Mathematically, the division of the quadrat/hull polygon into triangles creates one less interior edge when compared to the number of exterior hull edges. Using the interior triangle edges to determine a test points location generally permits the termination of the test faster when a point is interior to the hull edge, therefore significantly reducing the overall number of failed directed line tests.

The best overall quadrilateral point filter came from the Triangle fan model. Certainly, the topological arrangement of lines in the Fan contributed to its efficiency. The Triangle fan is a very fast algorithm when applied to the dynamic construction of the hull, because the hull topology provides a natural edge sorting mechanism for the implementation of the fan. Haines (1994) also points out that the faster Triangle fan algorithms for both general and convex polygons are those whose edges or areas are presorted, prior to the search. and that the it is a good algorithm when the majority of points are interior to the polygons (ie. hull) edges, which implies that a non-uniform circular point distribution pattern would likely be the most expensive hull to compute. Another important point is that the division of the point set into four parts, one for each quadrilateral edge, usually results in the evaluation of four smaller polygons, which provides the Triangle Fan with its "optimal" configuration, as it is the most efficient on polygons with fewer edges. Whether these facts result in the maximization of topological efficiency was not proven, however, it is reasonable to assume that the fan is faster than the grid based algorithms when there are fewer polygon edges because of these inherent characteristics.

2.7 Bibliography

Akl, S.G. and G.T. Toussaint. 1978. A fast convex hull algorithm. Inform. Proc. Lett. 7(5): 219-222.

Badouel, D. 1990. An efficient ray-polygon intersection. In *Graphics Gems*, A. Glassner, Ed. Academic Press, Boston. pp. 390-393.

Brassel, K.E. and D. Reif. 1979. A procedure to generate Theissen polygons. Geog. Anal. 11(3): 290-303.

Chan, T., J. Snoeyink and C.K. Yap. 1995. Output-sensitive construction of polytopes in four dimensions and clipped Voronoi diagrams in three. Proc. ACM-SIAM Symp. on Discrete Algorithms., pp. 282-291.

Dehne, F. and R. Klein. 1997. The big sweep: On the power of the wavefront approach to Voronoi diagrams. Algorithmica. 17: 19-32.

Graham, R.L. 1972. An efficient algorithm for determining the convex hull of a finite planar set. Inform. Process. Lett. 1: 132-133.

Green, C. 1993. Simple, fast triangle intersection. Ray Tracing News 6(1).

Hacker, R. 1962. Certification of algorithm 112: position of point relative to polygon. Comm. ACM. 5: 606.

Haines, E. 1994. Point in polygon strategies. In *Graphics Gems IV*, P.S. Heckbert eds., Academic Press Inc., Cambridge, MA., pp. 24-46.

Maus, A. 1984. Delaunay triangulation and the convex hull of n points in expected linear time. BIT. 24: 151-163.

O'Rourke, J. 1995. Computational Geometry in C. Cambridge Univ. Press, NewYork, N.Y.

Ohya, T., M. Iri and K. Murota. 1984. A fast Voronoi diagram algorithm with quaternary bucketing. Infor. Proc. Lett. 18: 227-231.

Preparata, F. P. and S. J. Hong. 1977. Convex hulls of finite sets of points in two and three dimensions. Comm. ACM. 20: 87-93.

Schneier, B. 1994. N-P Completeness. In *Dr. Dobbs Journal*, Sept., Miller Freeman Production. San Mateo, CA., pp. 119-121.

Schorn, P. and Fisher, F. 1994. Testing the convexity of a polygon. In Graphics Gems IV, P.S. Heckbert, Ed. AP Professional, Cambridge: 7-15.

Shimrat, M. 1962. Algorithm 112: Position of point relative to polygon. Comm. ACM. 5: 434.

Tsai, V.J.D. 1993. Fast topological construction of Delaunay triangulation's and Voronoi diagrams. Comp. & Geosci. 19(10): 1463-1474.

Wenger, R. 1997. Randomized Quickhull. Algorithmica 17: 322-329.

3. Voronoi Tessellation

3.1 Introduction

In this chapter, I focus on the design and implementation of a randomized divide and conquer algorithm for diagram creation that will provide a complimentary data structure for the planned statistical methodology. My planned statistical method follows that proposed by Okabe and Miki (1984) for conditional locational interdependency. Briefly, the statistical analysis requires that the Voronoi tiles are divided up into triangular segments, with the generator point acting as the anchor for each tile segment. This allows for rapid geometric classification of the triangle segments and the integration of random areas for statistical purposes. To compliment the statistical requirements I decided to attempt the creation of a randomized divide and conquer algorithm for Delaunay and Voronoi diagram construction, which in turn compliments the theory and techniques developed for the Quickhull model.

For the construction of tessellation diagrams, the determination and maintenance of the topology presents the most difficult problem to resolve (Spaccamela *et al.* 1996, Guibas and Stolfi 1985). This is due in part to the fact that for every point inserted into a Delaunay diagram, two additional triangles are created. This increase in triangle objects nonlinearily increases the search time for the next point insertion. For my algorithm I simply defined the topology of the tessellation through the use of triangle objects. All triangles are ordered in the same counter-clockwise direction, as tradition has it. Edge ordering is maintained during the insertion of new points by reordering the edges of the affected triangles. Point location is accomplished by the edge-paving technique, which uses the familiar directed edge test (Akl and Toussaint, 1978 and Guibas and Stolfi, 1985).

Even if the topological constructs are correct the model must still satisfy the geometric properties set out for each specific diagram type. The Delaunay diagram is defined by eighteen specific geometric properties. However, robustness and consistency in diagram computation can be derived through the coding of just three important properties. The first is the nearest neighbor property. It states that a triangle is Delaunay if and only if for each vertice of the triangle, the other two points are those points which are closest to this point than to any other in the dataset. The second property is the circumcircle property, which states that any point falling within a circle created from the three points of a triangle, i.e. the triangles circumcircle, defines a new nearest neighbor point for the triangle. The third property is called the cocircular property. If any four points from the dataset fall on the perimeter of the same triangles circumscribed circle, the triangles are unresolved. This means there can be no more than three points on the perimeter of a nearest neighbor circle and the circle itself must be empty. The Delaunay diagram is thus defined by determining each triangle's largest empty circumcircle. Geometric properties of the Delaunay and Voronoi diagrams are thoroughly treated by Okabe *et al.* (1992).

The model I propose consists of the following steps:

1. Division of the hull into a series of triangles.
2. Creation of a one dimensional array representing the triangle section that each point in the dataset falls into.
3. Insertion of points into their respective triangle sections and the swapping of redundant edges to form individual Delaunay triangle sections.
4. Conquer the problem by merging the triangle sections together and updating the necessary edge topology.

In the next section I describe the triangulation of the convex hull, its topology and the rational of the design. In section 3.3 I describe the creation of the one-dimensional array and the initial insertion of a point into the hull triangle sections, including the updating of the topological records. In section 3.4 I describe the insertion of the remaining points and the swapping of the triangle edges. The next section describes the final merging of the triangle segments to derive the final Delaunay diagram. In section 3.6 I describe the creation of the Voronoi tile objects and discuss both the duality of the diagrams along with the rational for the data structure used. Lastly I summarize the model developments. For consistency all examples throughout the chapter are derived from the same test dataset, which is listed in the appendix.

3.2 The Hull Triangulation Process

The overall aim in the hull triangulation process is to divide up the hull dataset into an ordered set of triangles with their edges oriented according to a standardized topological pattern. I begin the process by dividing up the hull into a sequence of triangles since this structure already presents itself as a natural triangular fan. Consider selecting one vertice of the hull and using it as an anchor point. From this point connect the next two points in the hull sequence so that together the three points form a closed triangle. Repeat this process in sequence until all triangles are formed and the result is as shown in Figure 3.2.1.

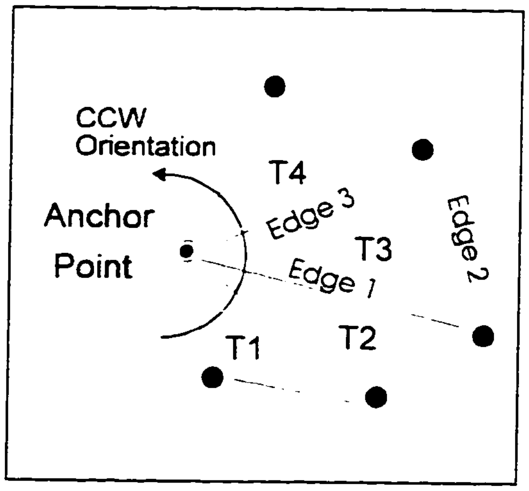


Figure 3.2.1. The initial hull triangulation.

The vertices and edges of each triangle are numbered in counterclockwise order from the anchor. The edges of each triangle are numbered in counterclockwise order beginning from the anchor point. Thus each triangle has its third edge adjacent to the next triangles first edge and so forth. Hull edges are represented with zeroes to permit filtering during the point insertion process. The record structure, as given in Table 3.2.1, represents this pattern. The remaining three numbers are the record numbers of the adjacent triangles. For example, consider triangle one, it is record number one in the dataset. Initially the record has three x,y vertices, named A,B and C. Its' first edge is a hull edge, hence the adjacent triangle AB field is numbered zero, so too is edge BC, however, edge CA is bordered on by triangle two, thus the adjacent triangle CA field has the number two in it. To topologically match the dataset the second record has a one in the adjacent triangle AB field designating that the first edge of triangle two is matched with the third edge of triangle one. Therefore, positioning of the adjacent triangle record numbers within a triangles' record is significant. Similarly for the remaining triangles in the set. The design of the edge topology in this manner provides a consistent framework for comparing point locations with triangle edges. Program control structures can filter out the redundant hull edges and target the testing of the remaining ordered edges. As I shall describe, this topology also reduces the number of edge updates and promotes rapid point insertion using the proposed edge paving process.

Initial Hull Triangle Records									
	Vertice A		Vertice B		Vertice C		Adjacent Edges		
Record	X	Y	X	Y	X	Y	AB	BC	CA
1	0.2939	18.4395	5.7082	8.5194	11.9234	1.0686	0	0	2
2	0.2939	18.4395	11.9234	1.0686	25.6483	1.5545	1	0	3
3	0.2939	18.4395	25.6483	1.5545	31.5235	11.6695	2	0	4
4	0.2939	18.4395	31.5235	11.6695	30.9537	20.8697	3	0	5
5	0.2939	18.4395	30.9537	20.8697	27.3888	31.2156	4	0	6
6	0.2939	18.4395	27.3888	31.2156	0.9977	31.3476	5	0	0

Table 3.2.1. Record structure for initial hull triangulation.

To complete the first step in the process each record is written to a temporary file and the adjacent edges are updated to zeros, which represents the fact they are all outer edges. This creates the divide portion of the strategy.

3.3 Pre-Insertion Processes

In this section, I describe the process of creating the one dimensional array and describe the process of inserting the first point into a triangle section, as this prepares the triangles for the rest of the insertion processes. To begin the creation of the one dimensional array. I use the hull triangulation much like the formation of the convex hull edge. By testing a points position relative to the inner edges of the hulls' triangles I can determine which triangle encloses the point. First, I randomly generate a record number in the range of one to the total number of hull triangles and then retrieve the triangle record at this location. Next, I select the first non-hull point from the input data file and test its' location with respect to the edges of this random triangle by applying the directed line test as described by Akl and Toussaint (1978).If the point falls interior to the triangle I must test the triangles opposite side to ensure it is enclosed. Since my aim is to determine in which direction I must go to find a points' enclosing triangle the edge test is set up to return the adjacent triangles record number, whenever the test point is exterior to the edge being tested. I would then test the adjacent triangles untested edges in counterclockwise order. Recall that the edges are interconnected so the if a point is exterior to one triangles edge it is known that it is interior to the adjacent triangles matching edge, therefore there is no need to test the edge again. If the test is exterior to an given edge I repeat the test on the next adjacent triangle, until the enclosing triangle is found. This is the edge paving technique (Figure 3.3.1). Each point record has the index number appended to it. This allows the program to select the appropriate record set to begin the search for the points enclosing triangle.

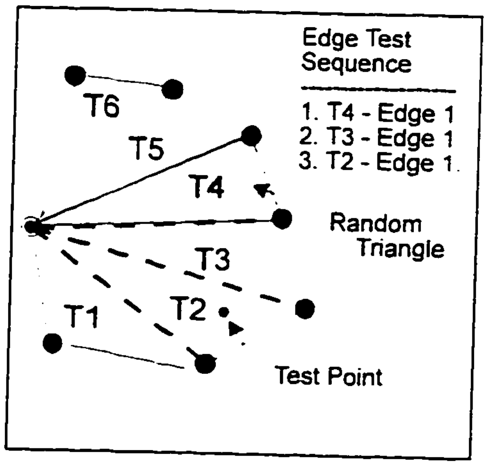


Figure 3.3.1. The edge paving process. Dashed lines are those used in the directed line tests.

Using the point index, I begin the insertion process by first dividing up the initial triangle segment into three new triangles. The triangle is split up into three new triangles and numbered in a counterclockwise fashion (Figure 3.3.2). Each new triangle begins at the test point location and revolves counterclockwise to its next two vertices. The original triangle record is updated with the new coordinates and its respective adjacent triangle numbers. Table 3.3.1 provides an example of the point (10.4219, 10.9027) inserted into triangle record two, using the same records from Table 3.2.1. Notice that the original triangle record has become triangle two in the record set. This standardization of the updating procedure is an attempt to maintain the position of the hull edges in the record set. If the first three records in each of the segment record sets are those triangles which have segment edges and they are arranged in this counterclockwise order then the process of merging the segments together will be straight forward, otherwise I will require to find the merge records and match them to the records from the adjacent segment sets.

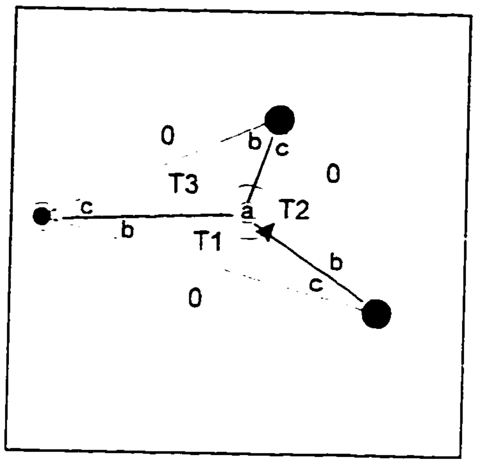


Figure 3.3.2. Triangle splitting geometry.

Hull Triangle Records									
	Vertice A		Vertice B		Vertice C		Adjacent Edges		
Record	X	Y	X	Y	X	Y	AB	BC	CA
1	10.4219	10.9027	0.2939	18.4395	11.9234	1.0686	3	0	2
2	10.4219	10.9027	[illegible]	[illegible]	25.6483	[illegible]	1	0	3
3	10.4219	10.9027	25.6483	1.5545	0.2939	18.4395	2	0	1

Table 3.3.1. Insertion of initial point into the triangle segment records.

Therefore, maintaining the topological order here ensures that future redundant operations are omitted and the record sets are stored in a fixed format permitting an evaluation of their correctness.

3.4 Edge Swapping Routines

I have thus far created the convex hull from the data file, triangulated the interior of the hull space, divided the triangle segments into separate files, inserted the initial points into each triangle segment and updated its records. In this section I will describe the insertion of the remaining points into the triangle segments and describe the swapping procedure in detail. As previously stated, the geometry of the triangles must correctly abide by the properties of the Delaunay diagram and to ensure that they are I will filter the redundant edges using the radius test. First I will describe the applicable theoretical properties, then describe the tests used to ensure the properties are correctly met and lastly I will explain the geometry of the edge swapping procedure.

All Delaunay triangulation's follow the empty circumcircle and non-cocircular properties. The theorem states that any given triangle formed inside a convex hull is Delaunay, if a circle, having on its' perimeter the vertices of the given triangle (i.e. the triangles circumcircle), does not enclose any other vertices in the triangulation and does not have on its perimeter any other points from the triangulation (Okabe *et al.* 1992). Figure 3.4.1 shows the simple geometry of these properties.

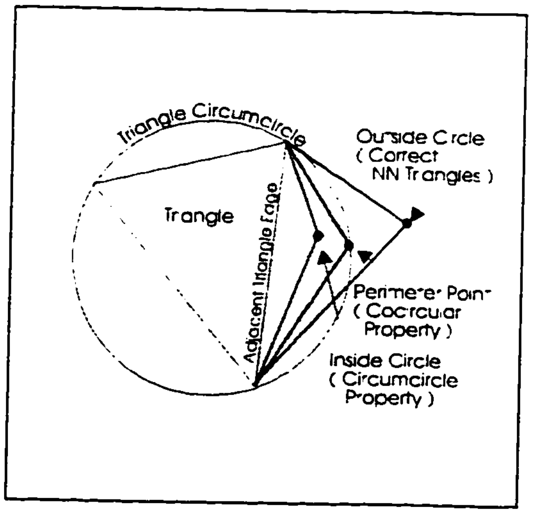

Triangle Circumcircle
Outside Circle (Correct NN Triangles)
Triangle
Adjacent Triangle Edge
Perimeter Point (Cocircular Property)
Inside Circle (Circumcircle Property)

Figure 3.4.1. Delaunay circle properties

The insertion of a point into the triangle segments begins as it did in the previous section, with the generation of a random number in the range of one to number of records, in this case three. I next edge pave to the enclosing triangle and use the enclosing triangles circumcircle properties to determine if the point is inside the triangles circle boundary. The centroid of the triangles circumcircle is calculated using a standard simultaneous set of equations for a circle with three points on its perimeter. A mathematical example of solving this form of equation can be found in Washington (1970). I used a modified version of Johnson's (1987, pp. 85-86) center program written in C. It uses triangle objects as opposed to individual points and it also error checks the values prior to execution of the divisions. Once the centroid is calculated I use the standard distance formula for calculating the distance from the data point to the triangles centre. This radius (r2), is compared to the triangles radius (r1), if r2 is smaller then the triangles edges are flagged for swapping (Figure 3.4.2).

The swapping technique rotates the vertices of each triangle in a counterclockwise manner. This creates two new triangles with the same matched edge, but different vertice values (Figure 3.4.3). The matching process does not have to update each and every edge

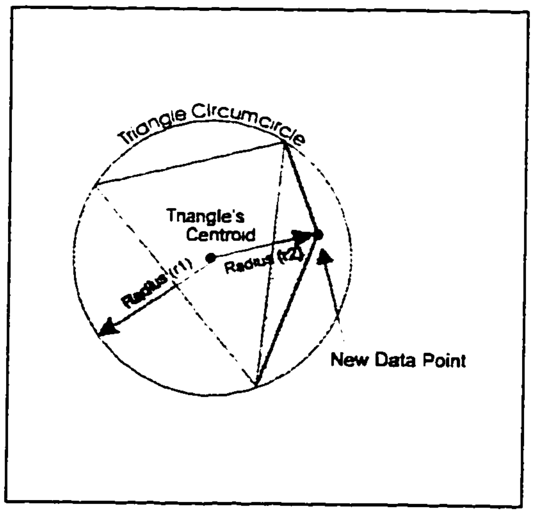


Figure 3.4.2. Radius test geometry.

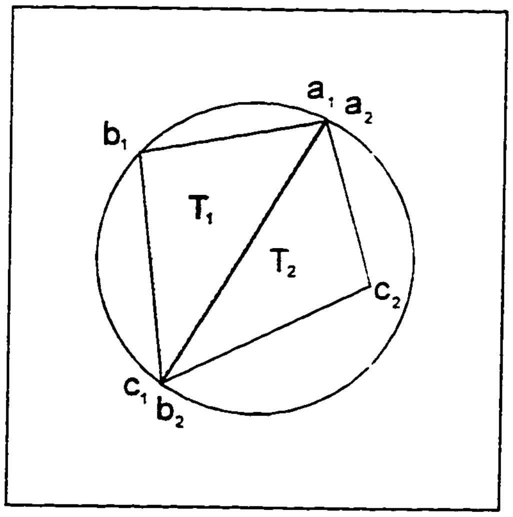


Figure 3.4.3. Radius test geometry.

adjoining the new triangles, since some of the adjacent record associations are still valid. The rotation of the matching edge maintains the edge order, hence both triangle records still crossmatch to one another as before . However, each triangle now has one edge from the other triangle. The matching adjacent triangle of each of these edges no longer points to the correct triangle record and must therefore be updated. The updating of these two edges is a matter of selecting the appropriate adjacent triangle record numbers, which are part of the original record structure, and updating them. The thick lines of Figure 3.4.4. are an example of the post swapped triangle edges. If a triangle is marked for edge swapping the adjacent triangles are also checked to determine if they still meet the Delaunay criteria. The geometry of this swapping procedure is similar to the method outlined Tsai (1993). An example of a point inserted into the initial triangle segment is provided in Figure 3.4.5. along with the

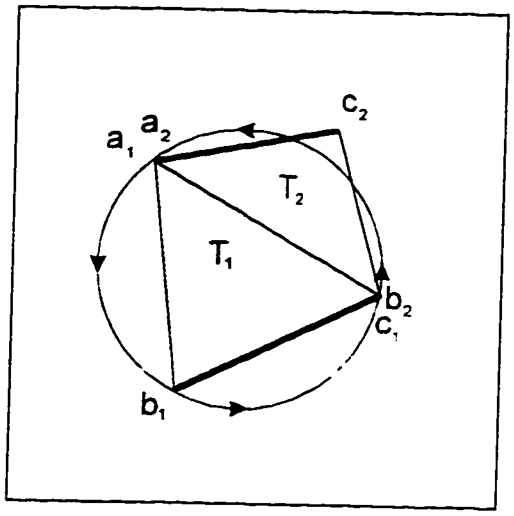
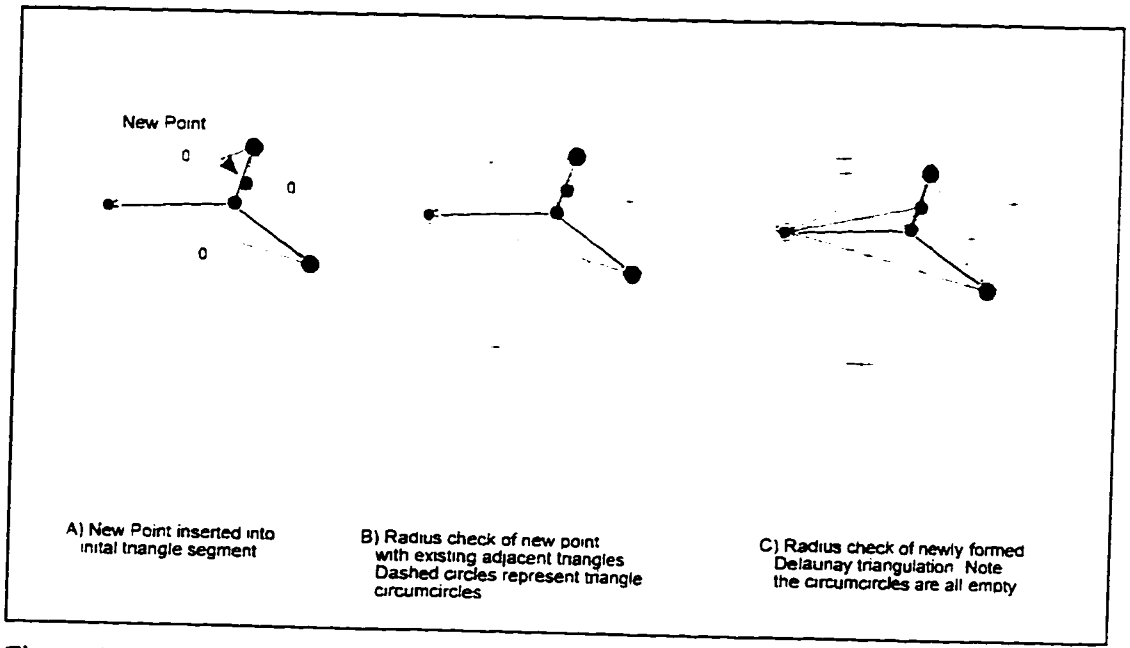


Figure 3.4.4. Post-swap Triangles


New Point
A) New Point inserted into inital triangle segment
B) Radius check of new point with existing adjacent triangles Dashed circles represent triangle circumcircles
C) Radius check of newly formed Delaunay triangulation Note the circumcircles are all empty
Figure 3.4.5. Geometry of the edge swapping sequence.

updating of its records (Table 3.4.1).

Hull Triangle Records									
	Vertice A		Vertice B		Vertice C		Adjacent Edges		
Record	X	Y	X	Y	X	Y	AB	BC	CA
1	10.4219	10.9027	0.2939	18.4395	11.9234	1.0686	4	0	5
2	16.3721	8.20854	11.9234	1.0686	25.6483	1.5545	1	0	3
3	16.3721	8.20854	25.6483	1.5545	0.2939	18.4395	2	0	4
4	16.3721	8.20854	0.2939	18.4395	10.4219	10.9027	3	1	5
5	16.3721	8.20854	10.4219	10.9027	11.9234	1.0686	4	1	2

Table 3.4.1. Triangle segment record set .

The result of the swapping technique is the Delaunay diagram for each triangle segment. All that is required to conquer the problem is the merge step.

3.5 Merging and Checking

The merge step is quite straightforward, since the topology for it is built into the model. I first selected the first set of triangle segment records and selected the third record from the set, which matches the first record in the second set. These triangles were then evaluated in the same manner as for the insertion of a new point in a triangle segment. If the triangles were not Delaunay, they rarely are, then the edges are swapped and the swapping procedure is continued until there are no longer any redundant edges between the segments. Once complete the records are merged into a single file for inspection. The resulting diagram from the merge step is the Delaunay diagram (Figure 3.5.1).

A close inspection of the diagram reveals that all triangles are indeed those with the largest empty circumcircles. I have drawn in a few circumcircles, in figure 3.5.1 to confirm the condition was met. As well I checked to ensure that the diagram had the correct number of edges and so forth using property D11 as stated in Okabe *et al.* (1992, pg.102). Property D11 is defined as follows: Given that n is the number of points in the dataset, if nt is the number of triangles in the network, ne is the number of edges in the network and nc is the number of vertices on the convex hull then the following equations hold:

$$nt = 2n - nc - 2, \tag{2.2.1}$$

$$ne = 3n - nc - 3. \tag{2.2.2}$$

Using these equations I should have 40 triangles and 64 edges for a dataset of 25 points, having eight points on the convex hull. A careful inspection of the Delaunay diagram confirms that the triangulation has the proper number of edges and triangles and is topological correct.

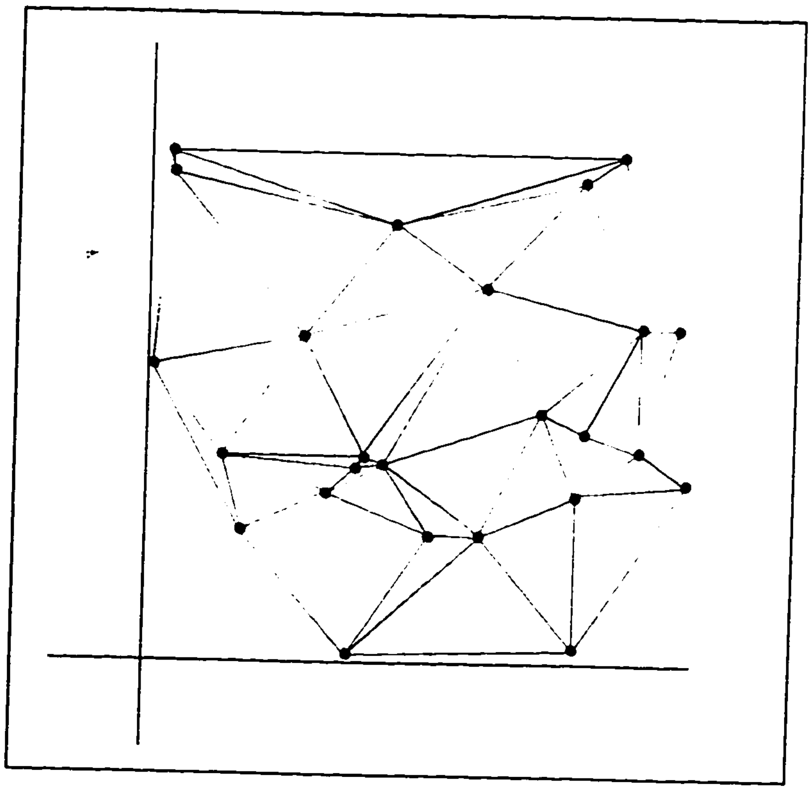


Figure 3.5.1. The Delaunay tessellation.

3.6 Voronoi Tessellation

The rational for the strict adherence to the fundamental geometric properties is from the realization that if the Delaunay diagram is not correct the mistakes there compound themselves when the Delaunay is used to develop the Voronoi. Since the Voronoi is the dual graph it serves as a final check for the Delaunay.

The Voronoi diagram is developed by simply connecting the circumcentres of the Delaunay triangles into closed loops around each of their nearest generator points. The connected circumcentre loops are formerly called Voronoi tiles (Figure 3.6.1). Voronoi tiles derived from Delaunay triangles with hull edges, do not create closed tiles. Without the formation of a boundary, the Voronoi is thus an infinite structure. The closed loops represent local areas of influence about the generator points. The tile boundaries represent the transition between two generator influences. An object, thus placed on a tile boundary experiences equal magnitudes of influence from the generator points on opposite sides, provided the plane is homogenous.

The record structure of the Voronoi is represented by an array of points, with the generator point as the first element in the array. This structure is not the most efficient representation of the diagram but it parallels the geometry used for the statistical process. Other representations such as the winged edge structure by Baumgart (1975) and quad edge structure by Guibas and Stolfi (1985) are far more efficient at the dual representation of the diagrams. Using the first element of the array (the generator point) as the anchor point the tile can be segregated into a series of consecutive

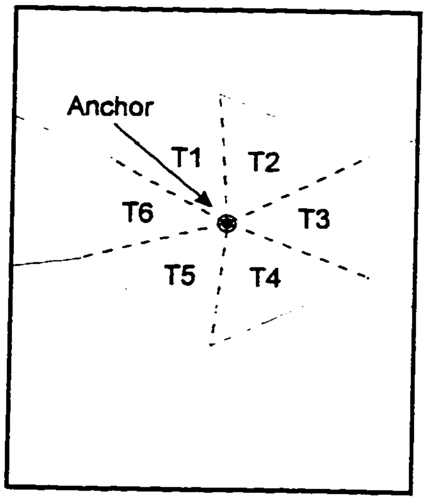


Figure 3.6.1. Tile triangulation.

triangles (Figure 3.6.1). Each triangle shares an edge with both the previous triangle and the next triangle in the list. This parallels the geometry provided in the paper by Okabe and Miki (1984). Consequently, it is easy to see that the use of triangle objects lends itself to the representation of both geometric structures, which is ideal for execution using the object oriented nature of C++ programming. The duality of the tessellation geometry is provided in Figure 3.6.2.

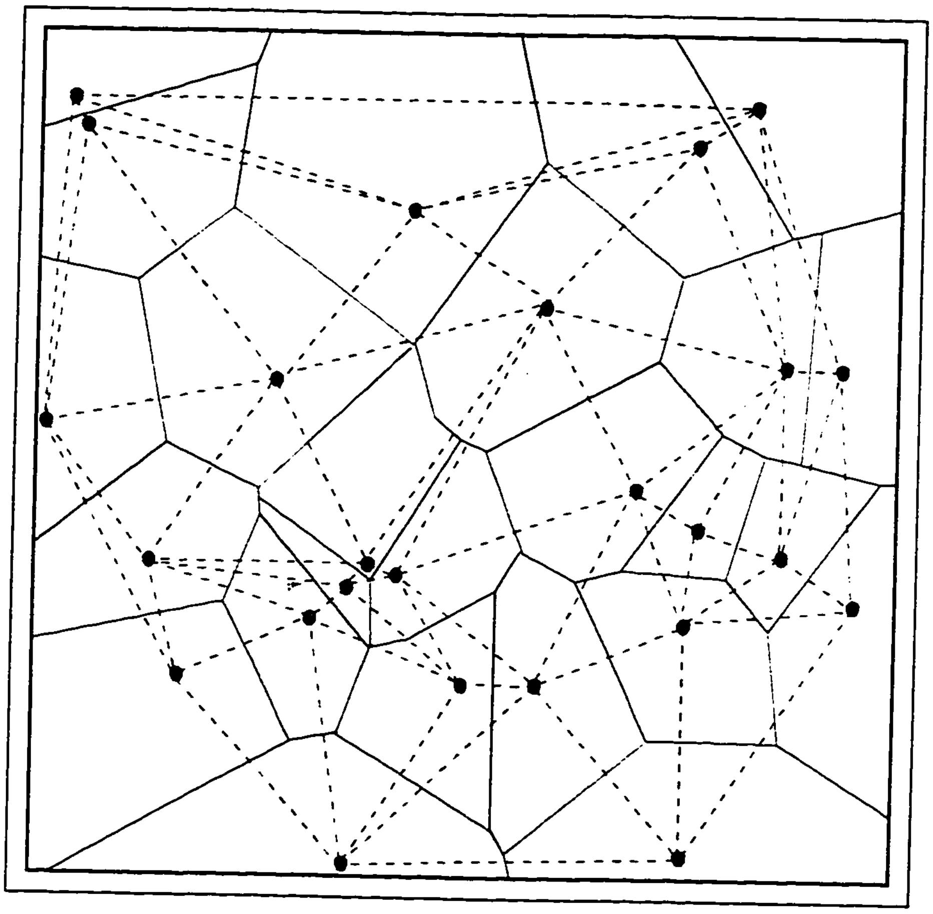


Figure 3.6.2. The Voronoi diagram (solid lines) and the Delaunay diagram (dashed lines) from the test data.

3.7 Summary

To objective of developing a Voronoi diagram using a randomized divide and conquer strategy was accomplished. The development of a Delaunay diagram as described appears to provide a consistent triangulation. I say this with some hesitation since it is difficult to deal with the wide variety of possible numerical errors. However, I have incorporated an epsilon tolerance into the calculation of the radius values in order to deal with the possibility of cocircularity. Unfortunately, this slows the program down slightly.

To summarize the process, I calculated the Delaunay tessellation from a convex hull by dividing out the hulls planar region into triangular segments. These segments are an attempt at dividing up the point dataset and will do so if they are randomly distributed. The division allows for a reduction in search time to locate a points enclosing triangle. From this point forward the process evaluates the inner radius distances between adjacent triangles to determine which points in the triangulation are the insertion points nearest neighbors. The affected records are then updated and the program resumes the selection of the next point in the dataset. If the triangle that encloses a point has an outer edge as one of its edges then the record number of the enclosing triangle remains the same. This provides an interlocking topology between the triangle segments which functions to simplify the merging together of the triangle segments to form the final record set. The sequential merging of the segments in counterclockwise order ensures that intermediate topology is correct prior to the merging of the next segment. The aim of this is to remove the need for the computer program to perform a final check though all the records.

The theory of the sequential ordering of geometric primitives is a fundamental part of the divide and conquer algorithm presented here. The verification of this theory will be addressed in future work. Also, testing of the models complexity will be completed at a later time.

3.8 Bibliography

Akl, S.G. and G.T. Toussaint. 1978. A fast convex hull algorithm. Info. Proc. Lett. 7(5): 219-222.

Baumgart, B.G. 1975. A polyhedron representation for computer vision. Proc. AFIPS Natl. Comp. Conf. 44: 589-596.

Guibas, L. and J. Stolfi. 1985. Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams. ACM. Trans. Graphics 4(2): 74-123.

Johnson, N. 1987. Advanced graphics in C. Osborne McGraw-Hill, Berkeley, Calif. U.S.A.

Okabe, A. and F. Miki. 1984. A conditional nearest neighbor spatial-association measure for the analysis of conditional locational interdependence. Env. Plan. A. 16: 163-171.

Okabe, A., B. Boots and K. Sugihara. 1992. Spatial tesselations : concepts and applications of Voronoi diagrams. John Wiley and Sons Inc. NewYork, N.Y.

Spaccamela, A.M., U. Nanni and H. Rohnert. 1996. Maintaining a topological order under edge insertions. Info. Proc. Lett. 59: 53-58.

Tsai, V.J.D. 1993. Fast topological construction of Delaunay triangulations and Voronoi diagrams. Comp. & Geosci. 19(10): 1463-1474.

Washington, A. J. 1970. Basic technical mathematics with calculus. Cummings Publ. Co., Menlo Park, Calif.

4. Statistical Tessellations of Moose Point Locations

4.1 Introduction

Statistical tessellation derived its roots during the second world war when the focus of interest was the prediction of damage zones created from aerial bombings. The initial mathematics were developed by Robbins (1944), Robbins (1945) and Bronowski and Neyman (1945). Garwood (1947) then used these investigations to derive equations for calculating the variance of disks randomly distributed in the plane. He viewed the problem in its true inverse, calculating the variance of the regions within which the disks were placed and not the disks themselves. Despite the fact that today we calculate the variance of the disks and not the regions that they reside in, these investigations still stand as the cornerstone for the mathematics used in this form of spatial statistics.

Tessellation statistics of this form are more often used for point pattern analysis. Boots (1974) used the Delaunay diagrams to test if a given random point pattern was significantly different from that expected from a homogenous Poisson process. He determined that angles within the diagram could be used to distinguish the differences but warned that it may be too insensitive for other point distribution patterns. In contrast, Lenz (1979) used the Voronoi diagrams to test the pattern of personal robberies in the county of Milwaukee. He applied Thiels' (1967) measure of redundancy to note the differences in the locations of the robbery victims homes to the actual personal robbery locations. Using Voronoi diagrams allowed Lenz to map the victims "journey to crime". There are several other statistical examples using tessellated diagrams for things ranging from park proximity analysis to road network analysis, but the abundance of work revolves around point pattern analysis.

Although the above examples could be applied to wildlife analysis I chose to use the methods developed by Okabe and Miki (1984), as the statistical basis for the formulas best fit my dataset. These methods are based on the principles of conditional probability theory, which estimate the probability of an event occurring when it is known that another, perhaps related event, has already occurred. This can be translated into determining the probability of moose being influenced by the existence of related fixed features, like lakes. We can assume that the placement of the lakes is due to the topography of the land and not due to the location of moose. On the other hand, moose locations are likely influenced by the location of the lakes and quite likely they are distributed in proximity to a few specific lakes. In other words the moose locations are "conditioned" by the lake locations. In this chapter I investigate the use of statistical tessellation to determine if I can isolate some of the natural landscape features which influence moose distribution patterns.

In section 4.2, I provide a brief review conditional probability theory as it relates to tessellation geometry. In section 4.3, I summarize the necessary background for Okabe and Mikis' spatial statistic Rc. Along with it I provide an empirical calculation of the probability density function and discuss the disk and triangle geometry. In the third section, I calculate the influence of lakes and rivers on the distribution of moose and in the chapter summary I highlight the important parts of the calculations and briefly discuss my findings.

4.2 Probability Theory

We begin this discussion by outlining the general theorem of total probability. The following paragraphs and the subsequent diagram are excerpts from Walpole (1982). The theorem states that if the events B1, B2..., Bk constitute a partition of the sample space S such that P(Bi) > 0 for i = 1,2,3..., k, then for any event A of S:

$$P(A) = P(B1)P(A|B1) + P(B2)P(A|B2) + P(B3)P(A|B3)\ldots + P(Bk)P(A|Bk).$$

Which by probability theorem is identical to;

$$A = (B1\Omega A) \cup (B2\Omega A) \cup (B3\Omega A)\_\cup (Bk\,\Omega A).,$$

where Ω represents the intersection between B and A and U represents the union. Spatially this is represented by Figure 4.2.1. If the shaded area A represents an animals home range and the B's represent forest cover types then the probability that an animal is found within the home range, A and in the B1 cover type can be calculated using the following;

$$P(A|B1) = P(A\Omega B1) / P(A) = (A\,\Omega B1) / B.$$

Therefore, the probability of a point falling within the home range within the B1 cover type is proportional to the area occupied by A within B1. The same concept is applied to the Voronoi tiles. We generate a series of disks of a random radius and place them centered on the tile generators. These are conceptually the same as the home range. Then I ask what is the probability that a random event (moose location) will occur within a disk of radius r centered on the tile generator. The probability is exactly equal to the equation for total probability. Unfortunately, it is not easy to calculate the probability of B since the disk-in-tile geometry creates irregular shaped areas. To exactly calculate the area of these shapes it is traditional to turn to integral calculus for a solution. This will be pursued in next section.

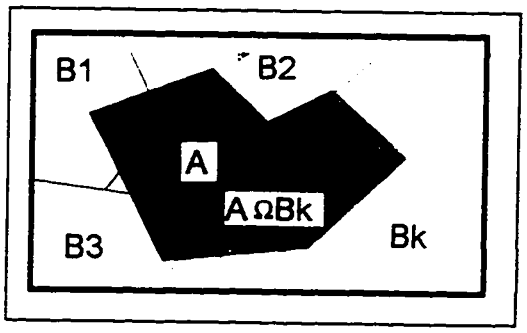


Figure 4.2.1. The total probability sample space.

Another difficulty is the determination of a non-biased boundary. The unit sum constraint states that total of the probabilities of the sample space must equal one, Aitchison (1986). Since the boundary, which is my sample space, is not clearly defined I know that the unit sum constraint is likely violated, however, an artificial selection of the boundary location would likely bias the probability space leading to an enlarging or shrinking the intersecting spaces. This is the general problem with the standard nearest neighbor analysis (Pielou 1967) and most quadrat based sampling. This is important since for infinite diagrams, like the Voronoi, the boundary must somehow be defined without biasing the probability. It is not clear how Miki derived the boundary for his example, however, I found that the boundary fit the extents of the dependent variable, being the retail stores in his example. If I consider that a non-biased estimate of the true boundary location would be one that on average estimated the true boundary location then using the extents of the observations would not provide a good estimate, because, within the context of the Voronoi diagram, there is no way of knowing were it could or should be. Miki's solution this leaves us with a biased boundary solution, since using the extents of the dataset will artificially enlarge the total sample space. Therefore I have developed a non-biased boundary solution which sets the stage to define the example problem.

If the moose locations are "conditioned" by a given fixed feature, I am testing the conditional probability that the spatial distribution of the moose (M) are being influenced given the spatial arrangement of the fixed locations (F), thus I state the problem as follows;

$P(M|F) = P(M \Omega F) / P(M)$, if $P(M) > 0$; where Ω is the intersection of M and F.

In reverse I have,

$P(F|M) = P(F \Omega M) / P(F)$, if $P(F) > 0$; where Ω is the intersection of M and F.

If the two are mutually exclusive then,

$$P(F|M) = P(F) \text{ or } P(M|F) = P(M).$$

Essentially, I am asking whether the fixed features are in some way related to the distribution of the moose or if it they are truly independently distributed. Logically, if the fixed features are the centroids of lakes I can be quite sure that they are independent of the moose, however, I could choose another, perhaps related distribution like the point locations of a local wolf pack. In this case, the techniques would require some modification to account for the "unconditional" relationship between the distributions (see Lee 1979). Note that using the dependent variable to create the Voronoi diagram, instead of the independent, enables me to test the relative strength of the reverse conditioning. This is important to check especially if there is cause to believe that the two distributions may be independent of one another. Whether this reversal procedure actually determines independence is, at present speculative.

4.3 Mathematics and Geometry

In theory the value of the nearest neighbor association measure, $\overline{Rc}$ is used to determine if the observed point locations of moose are closer to the natural features (called generator points) or farther away in comparison to a randomly distributed set of locations. The calculation of the randomly generated set of distances is standardized across the Voronoi diagram to include each tile within the diagram. In this way, a disk of radius r distance centered on each tile generator provides a means to divide up every tile in such a way that I can calculate the probability of a point being r distance from the generators. The summation of each set of radius probabilities provides an entry to form a customized probability density function for this diagram, from which I can compare the average point to generator distance of the observations to determine if the moose are closer to the generators or farther away than would be expected from a normal random distribution. Therefore the Voronoi tiles have a two-fold role to play. They are first instrumental in the calculation of the expected value of the random variable r and secondly they enable a rapid means to locate and measure an observations closest feature point.

In keeping with the notation of Okabe and Miki (1984), the value of the nearest neighbor association measure, $\overline{Rc}$, is calculated using the following formula:

$$\overline{Rc} = \Sigma\, \bar{r}_A / \mu,$$

where $\bar{r}_A$ is the average observed distance and μ is the mean of the random variable r. Given that r is a random variable independently distributed in the plane, then the calculation of $\overline{Rc}$ determines if the moose are independently distributed within the study area or whether they are influenced by the distribution of the natural features. If $\overline{Rc} < 1$ the moose are more closely associated with the generators than if they were randomly distributed, otherwise if $\overline{Rc} \geq 1$ I can conclude that the moose generally avoid these features. The calculation of $\bar{r}_A$ is straight forward, just take sum the distances of each location to their respective nearest generator point and divide by the number of locations. Unfortunately, the calculation of μ is slightly more complex. The formula for μ is as follows;

$$\mu = E(r) = \int_0^{r^*} r \, dF(r),$$

where r* is the maximum distance from a tile generator to a tile boundary across the whole tile set and r is a random variable. Typically the calculation of μ is achieved by overlaying a disk of random radius r onto each tile, aligning the disks' centroid with the generator. The circular disk is applied to the tile for the purposes of dividing the probability space of the tile evenly from the tile generator. Recall from the previous section that P(A)P(A|B1) is the probability of a random point falling within A given B1. F(r) parallels this probability, except that the home range A is replaced with a disk and the cover type B1 is replaced with a Voronoi tile. Using this geometry I can integrate the areas for randomly sized disks and subsequently solve for μ. In the next paragraph I give a preliminary description of the disk and tile geometry which will serve as a primer for the calculation methods which follow.

The geometry of the disk within a tile can be divided into a series of geometric shapes that resemble triangles and pie slices (or disk sectors). A Voronoi tile is easily divided into multiple triangles by connecting its generator to the vertices of the tile edges (Figure 4.3.1.). The triangles can then be categorized as acute or obtuse depending upon the angle formed by subtending the generator with the tile edge of that triangle (Figure 4.3.2.).

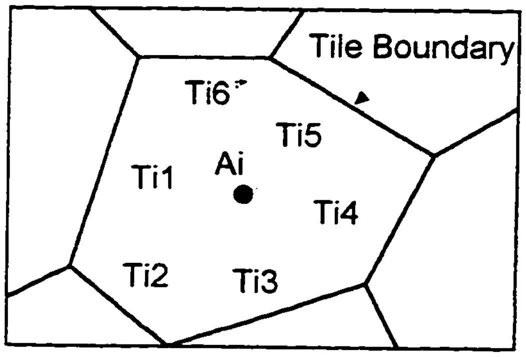


Figure 4.3.1. Tile triangulation.

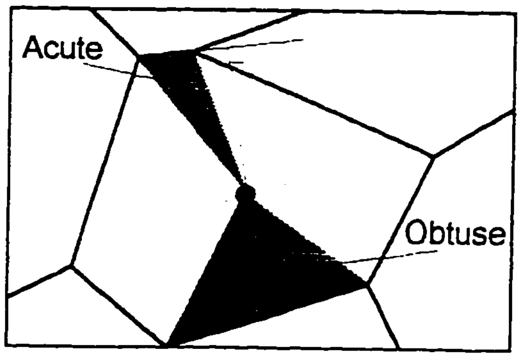


Figure 4.3.2. Acute and Obtuse triangles

We know that the when the disk is completely enclosed within the tile that the area is equal to Pi$(r)^2$, but due to the tessellation geometry I know that as the disk radius grows each disks' area of influence is bounded by the tile edges, which is the doubly hatched area in Figure 4.3.3, hence to derive the probability of a random point falling within the disk I must eliminate the parts of the disk that fall outside the tile edges. There are two distinct methods of calculating the areas of the clipped disks. The first is by integral calculus (Okabe and Miki 1984) and the second is by plane analytic geometry. I negate the calculation by plane analytic geometry and focus on the integral solution due to the fact there are references supporting the integral solution. Instead of reworking the formulas completed by Okabe and Miki (1984), I choose to provide an empirical example of an area calculation using one of the four proposed formulas.

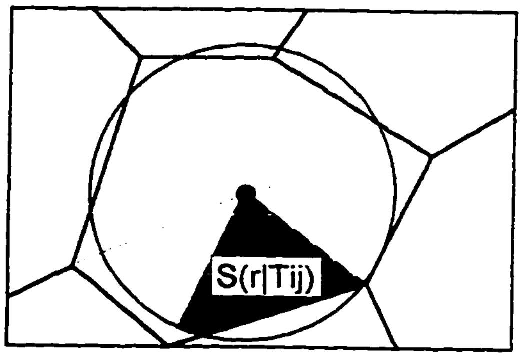


Figure 4.3.3. Disk in tile geometry.

In general, the integral methods developed by Okabe and Miki (1984) begins by focusing on the calculation of the probability density function F(r). F(r) of distance r from a random point to the nearest generator is given by;

$$F(r) = S(r)/S = \sum_{i=1}^{m} \sum_{j=1}^{q} S(r \mid Tij) / S, \qquad \text{Eqn. 4.3.2}$$

where m = the number of tiles, q = the number of triangles within a tile, r = the radius of the disk, Tij is the jth triangle in the ith tile and S is the area of the total sample space.

Consequently, the value of S(r|Tij)/S is identical to asking what is the probability of a random point falling into a set of tile disks of radius r. In common notation, it is usually referenced as P(r). I require to calculate each radius' probability in order to calculate the Expected value of r, otherwise called the mean or first moment. To proceed with the calculation of S(r|Tij) the edge lengths, the inner edge angles and the generator to edge height of triangle Tij is calculated (Figure 4.3.6 and Figure 4.3.7). Next I calculate the classification angle, $\angle$ AiVij+1Vij for triangle Tij. If it less than or equal to Pi/2 ,or 90^{0}. I set omega factor t to 1 as the triangle is obtuse, else omega is set to -1, as the triangle is acute. The omega factor is then incorporated into one of four equations. Which equation to choose depends upon the radius of the disk with respect to the length of the triangles edges and these are outlined in Okabe and Miki (1984). Once the areas for all the triangles in the sample space have been determined they are summed to provide a single entry for the density function F(r).

As the disk radius increases and the function is formed I know that the radius r is finite in size since the sample space is bounded on all sides. Likewise I know that the observations are at maximum, r maximum (r*) distance from any given tile generators since the extremes of these observations are what was used to form the study area boundary. Which brings us full circle to the calculation of $\bar{R}c$ and leads us to an empirical calculation of area using both calculation methods.

Example 4.3.1 The Calculation of Disk in Triangle Area.

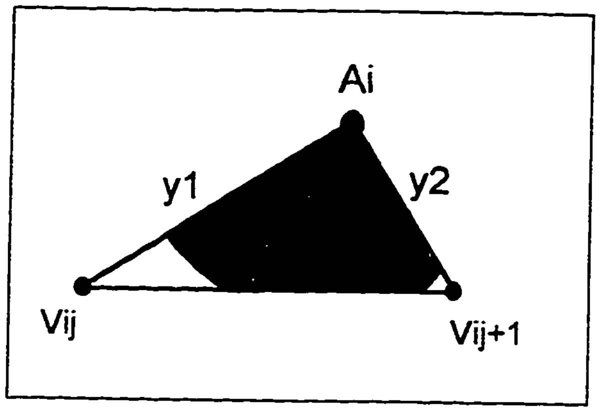


Figure 4.3.4. Disk area to be integrated.

Given the following coordinates of the vertices and generator of a Voronoi tile triangle as follows;

Ai.x = 4.0, Ai.y = 5.0, r = 3.5, Vij.x = 1.0, Vij.y = 2.0, Vij+1.x = 6.0, Vij+1.y = 2.0,

then to calculate the shaded area of Figure 4.3.6, I first calculate the length of each triangle leg using the distance formula,

$$y1 = \sqrt{(vij.x - a.x)^2 + (vij.y - a.y)^2} \qquad = \mathrm{sqrt}[(1 - 4)^2 + (2 - 5)^2]$$

$= 4.243$,

$$y2 = \sqrt{(vij_{+1}.x - a.x)^2 + (vij_{+1}.y - a.y)^2} \qquad = \mathrm{sqrt}[(6 - 4)^2 + (2 - 5)^2]$$

$= 3.606$,

$$y4 = \sqrt{(vij_{+1}.x - vij.x)^2 + (vij_{+1}.y - vij.y)^2} \qquad = \mathrm{sqrt}[(1 - 6)^2 + (2 - 2)^2]$$

$= 5$.

next I calculate the interior angles of the triangle using the Cosine Law and set the δT factor according to the value of Vij_{-1},

$$\cos Vij_{-1} = \frac{y2^2 + y4^2 - y1^2}{2(y2)(y4)}$$

$= 3.606^2 + 5^2 - 4.234^2 / 2(3.606)(5) = \cos^{-1}(0.5567) * 57.3$

$= 56.173$ degrees,

since 56.1733 is < than 90^0 $\delta T = 1$, next I calculate the angle Vij as follows;

$$\cos Vij = \frac{y1^2 + y4^2 - y2^2}{2(y1)(y4)}$$

$= 4.243^2 + 5^2 - 3.606^2/2(4.243)(3.606) = \cos^{-1}(0.7069) * 57.3$

$= 45.01^0$,

Using angle Vij, I calculate the triangle height using the Sine Law as follows;

$y3 = y1 * \sin(Vij) = 4.243 * \sin(45.01) = 3.0$

Next, I calculate the alpha angles 1 and 2 as follows;

$\alpha_1 = 180 - (90 + Vij) = 44.9^0$;

$= 44.9\ (3.141592654/180.0) = 0.7836$ radians

$\alpha_2 = 180 - (90 + Vij_{-1}) = 33.8^0$;

$= 33.8\ (3.141592654/180.0) = 0.5899$ radians

Next, I select the correct equation, as in Okabe and Miki (1984), by comparing the length of y3 to the radius, r of the disk, since r is greater than y3 and less than y2 the equation is as follows;

$$S(r|Tij) = r^2 / 2\{\alpha_1 - \Omega(\alpha_2) - [\arccos(y3 / r) + \Omega(\arccos(y3 / r))] \} + 0.5(y3)(r^2 - y3^2)^{0.5} + 0.5(\Omega)(y3)(r^2 - y3^2)^{0.5}.$$

$$= 3.5^2 / 2 \{ 0.5899 + 1(0.7836) - [\arccos(3.0/3.5) + 1(\arccos(3.0/3.5))] \} + 0.5(3.0)(3.5^2 - 3.0^2)^{0.5} + 0.5(1)(3.0)(3.5^2 - 3.0^2)^{0.5}.$$

$$= 6.125 \{ 1.3735 - 1.0822 \} + 2(2.704)$$

$$S(r | Tij) = 7.1925 \text{ units square.}$$

The calculation of the areas for a given radius is then summed and divided by the total study area to provide a single entry into the F(r) graph. As previously mentioned, from standard probability theory I know that the expected value of the random variable r (radius as before) is equal to;

$$E(\dot{r}) = E(r/\mu) = \Sigma\, r_i\, P(r_i) = \int_0^{r^*} r\, dF(r) = 1,$$

which implies that the variance is equal to;

$$Var(r) = \Sigma\, (r_i - \mu)^2\, P(r_i) = \frac{1}{n\mu^2}[\int_0^{r^*} r^2\, dF(r) - \mu^2].$$

which are identical to those provided by Miki in equations (7) and (8) respectively. This should help to clarify the basis of the probability relationships. Once the Var(r) and the mean of the observations has been calculated, all that is left is to calculate Miki's test statistic by the following formula;

$$\overline{Z} = \frac{\overline{R}_{C\,obs} - 1}{\sqrt{Var(R_C)}} = \frac{\overline{R}_{C\,obs} - 1}{StdDev(R_C)},.$$

which in essence normalizes the distribution of the observations so that they can be compared to the standard z table values. Note that the lower the value of the variance the less the disks sizes vary about their mean, which is the same as saying that the tiles are of very similar size and shape since the tiles essentials dictate the calculation of disk areas. Therefore, if the spatial parameters of the feature points dictate their respective tile shapes then I can predict that minor variations in the spatial positioning of the feature points or moose locations will effect the statistical outcome. It is this kind of sensitivity that we require, especially when it necessary to determine pre and post-effects of man-made disturbances, such as cut-blocks, on the distribution of a given wildlife species.

4.4 Moose Analysis

The location of the study area was in Northeastern Alberta, Canada. It is a rectangular region, approximately 63 km X 81 km in size, the coordinates for the study area in latitude and longitude are as follows: Lat. $112^0$00'00" to $113^0$15'00", Long. $55^0$34'00" to $56^0$10'00". This area is located within the Alberta Pacific Industries Forest management area. The region used for the example is a subregion of this study area. The region is dominated by peatlands. It has a prominent mixed wood forest and the river is a prominent feature geographical feature. By using the statistical methods formulated by Okabe and Miki (1984) I attempt to determine if moose locations are spatially associated with lake locations and a major river, the Athabasca. Centroids were generated using ArcInfo for the river and for each of the lakes. The moose point locations were VHF radio-triangulation data collected in April and May of 1995 from thirty-one collared female moose.

The first step in the analysis was the determination of the correct boundary. The correct boundary is made up all the closed tiles, because bordering tiles are closed by an artificially derived boundary I cannot use the calculations without biasing the outcome. Although this is not the process prescribed by Okabe et al. (1992) or in the paper by Okabe and Miki (1984), I know that the calculation of the statistic Rc is dependent upon the total study area and therefore I must use only the tiles whose complete area has been determined, these are the closed tiles.

To begin the analysis I first removed all moose locations that did not fall within a closed tile. This reduced the number of observations in all three months, however, proper collection of more feature information from the surrounding area can remove this problem, as this will have the effect of closing many of the otherwise unclosed tiles. The computer program then gave me all the information used to calculate the Rc value and then proceeded to calculate the final Z value. In the analysis of the May, June and July datasets, all but one of the enclosing tiles had moose locations within them. The results of the calculations are provided in Table 4.4.1. and a sample map of the June moose locations is provided in Figure 4.4.1.

Statistic	Month		
	May	June	July
Sample Size (n)	103	30	20
Avg. Dist. Obs. ($\bar{r}_A$)	8133.582	8433.175	8681.568
Conditional Index ($\bar{R}c$)	0.7912	0.8203	0.8445
Spatial Statistic ($\bar{Z}$)	-2.223	-1.9135	-1.655

Table 4.4.1. Statistics for the conditional interdependency of moose with lake and river centroids.

Using the tables for the standard normal areas, I found that the moose in May were spatially associated with the water locations at a significance level of 0.01. In April, June and July, the moose were still strongly associated with the lakes, however, they are not

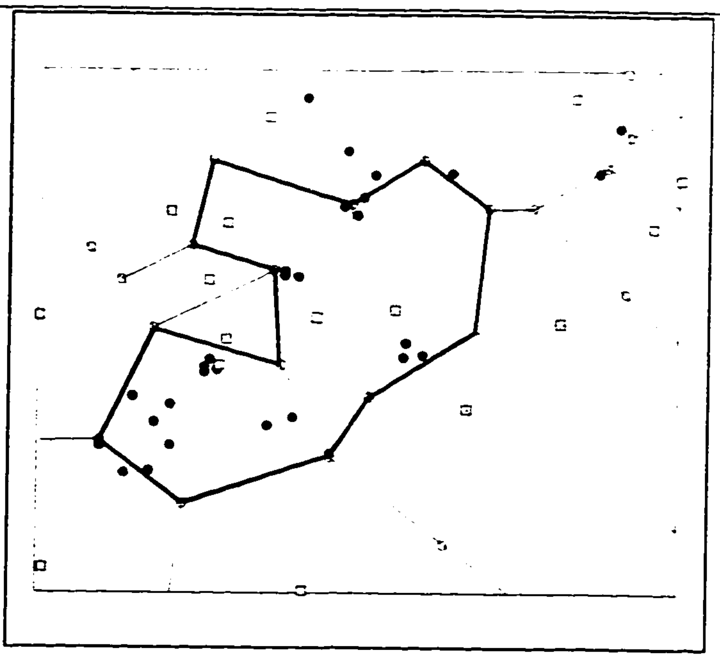


Figure 4.4.1. The April distribution of moose, (indicated by black circles) and the Voronoi tiles centered on the lake centroids (indicated by gray squares). The boundary is defined by the thick black lines.

significant at 0.01. There are significant biological connections to the definition of the boundary and these will be discussed in the final summary. Below I have included a graph for the function F(r), (Figure 4.4.2) and a table for the expected value, standard deviation for the disk areas (Table 4.4.2).

Disk in tile Values	
Statistic	
Number of disks	100
Std. Dev. (σ)	0.093910
Expected Value E(r)	10279.555

Table 4.4.2. Disk Values for Rc Calculations.

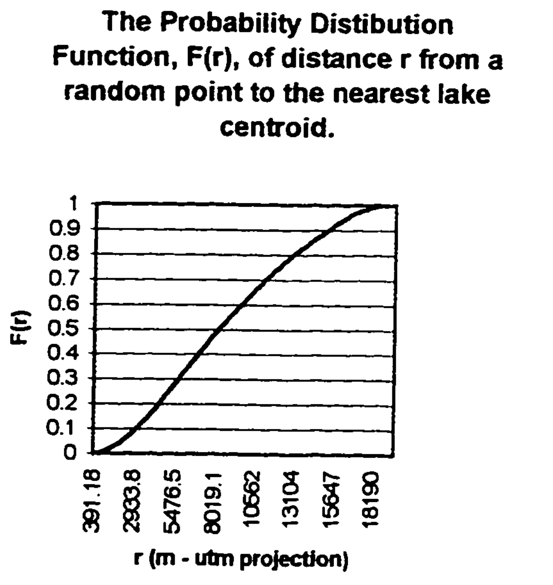


Figure 4.4.2. Distribution of the probability of random variable R.

4.5 Summary

Spatial probability theory is quite straight forward when one knows how to perceive the spatiality of the information. Viewed as the proportion of area within the study area, each tessellated tile readily becomes understandable as a sample space. Each space representing the homogenous area of influence for the locational observations. Provided the feature locations are independently distributed with respect to the observations I readily turn to conditional locational interdependency as a measure of their associations. In practice it should be noted that the calculation of Rc using the independent variable as the observations should completed. This would verify the conditional nature of the datasets.

Boundary effects are exactly taken into consideration if only closed tiles having observations within them are used. In my examples, I minimized the size of the boundary to fit the dataset exactly. This removes the arbitrary boundary bias present in the common quadrat and nearest neighbor statistics (Pielou 1961, Clark and Evans 1954).

The integral solution for the calculation of the disk in tile geometry is exact for any given triangle shape. The understanding of the integration is fairly straightforward, however, the method requires extreme care, due to the number of formulas and variables used.

The moose tessellation example does not provide a significant contribution to the our understanding of the effects water areas have on moose behavior; moose like riparian areas (DeGraaf and Rudis 1983). However, these findings hold much greater significance from the fact that they are derived from an unbiased spatial statistic.

4.6 Bibliography

Aitchinson, J. 1986. The statistical analysis of compositional data. Chapman and Hall. London, England.

Boots, B.N. 1974. Delaunay triangles: An alternative approach to point pattern analysis. Proc. Assoc. Amer. Geog., 6: 26-29.

Boots, B.N. and A. Getis. 1988. Point pattern analysis. Sage Publ. Inc. London. England.

Bronowski, J. and J. Neyman. 1945. Ann. Math. Stat. 16: 330-341.

Clark, P.J. and F.C. Evans. 1954. Distance to nearest neighbor as a measure of spatial relationships in populations. Ecology 35: 445-453.

Garwood, F. 1947. The variance of overlap of geometrical figures with reference to a bombing problem. Biometrika 34: 1-17.

DeGraaf, R.M. and D.D. Rudis. 1983. NewEngland wildlife: habitat, natural history and distribution. U.S. Dept. of Agriculture, Forest Service, N.E. For. Exp. Station, Gen. Tech. Rep. NE-108.

Lee, Y. 1979. A nearest neighbor spatial association measure for the analysis of firm interdependency. Env. Plan. A. 11: 169-176.

Lenz, R. 1979. Redundancy as an index of change in point pattern analysis. Geog. Anal. 11(4): 374-388.

Okabe, A. and F. Miki. 1984. A conditional nearest neighbor spatial-association measure for the analysis of conditional locational interdependence. Env. Plan. A. 16: 163- 171.

Okabe, A., B. Boots and K. Sugihara. 1992. Spatial tesselations : concepts and applications of Voronoi diagrams. John Wiley and Sons Inc. NewYork, N.Y.

Pielou, E.C. 1974. Population and community ecology: principles and methods. Gordon and Breach publ., New York, N.Y.

Pielou, E.C. 1969. An introduction to mathematical ecology. John Wiley publ., New York, N.Y.

Robbins, H.E. 1944. On the measure of a random set. Ann. Math. Stat. 15: 70-74.

Robbins, H.E. 1945. On the measure of a random set. II. Ann. Math. Stat. 16: 342-347.

Thiel, H. 1967. Economics and information theory. North-Holland publ., Amsterdam, Netherlands.

Walpole, R.E. 1982. Introduction to statistics. Macmillan Publishing, New York, N.Y.

5. Discussion and Conclusions

The current lack of an unbiased method to calculate the spatial associations between environmental variables and wildlife distributions lends significant weight to the methods proposed in this thesis. The combination of tessellation diagrams with conditional probability theory provides a means to develop hypothesis about the effects one spatial distribution has on another. It's inherent use of the independent variables spatial characteristics, such as juxtaposition, leads us away from the simplistic comparisons of random distributions as used by nearest neighbor analysis to the more substantial hypothesis based pattern analysis. Take for example the comparison of the patterns of moose with respect to environmental components at differing resolutions. At the landscape level we could predict that the spatial arrangement of moose with respect to its environment reflects a form of second order selection as proposed by Johnson (1980). A finer grained analysis could lead us to develop the home range or possibly the core areas used by the moose. Certainly, the conditional nature of the analysis and the interdependency of the tiles lends weight to these hypothesis. Also consider the use of the tile as a "patch" within the landscape. A simple analysis of the changing number of patches with respect to the occupancy by a species could measure the change of influence certain patches have as the landscape increases or decreases in its level of fragmentation. At what point does the fragmentation no longer provide adequate influence to maintain the stability of the distribution pattern of a species. Furthermore, tessellation provides a means to analyze any form of linear corridors, including road networks, pipelines, cutlines and so forth. It has the capacity to develop diagrams for any point, line or area feature within a landscape. Moreover, it is not restricted to using single feature types, as there are methods for tessellating multiple feature types, a parallel, in fact, to multi-variate analysis.

With respect to the research and development on the subject of tessellation diagrams provided in this thesis, it is hoped that the methods and explanations will serve as an introduction to a researcher new to this geometry. There is a wealth of literature on the subject and its depth is extraordinary, in both a mathematical sense and in an application sense.

The incorporating of the triangle fan algorithm into the Quickhull model dramatically improved its overall performance. Using triangles as the geometric primitive we find the natural sorted triangular fan enables a faster elimination of non-hull points using the Quickhull model versus that derived using Shimrats ray shooting technique. The primary reason was due mostly to the reduction in the number of failed directed line tests during the elimination of non-hull points along the hull path. We confirmed that Shimrats algorithm is still a very efficient point in polygon algorithm, as it will execute the model in linear time, however, the triangle fan is better suited to the design of the Quickhull model. Certainly the model could stand a more rigorous analysis of its distributional properties (i.e. its data density dependencies) which should lead to an even better hull model. It is important to note that this is not the only angle from which to approach the derivation of the Delaunay diagram, as there are several incremental approaches which do not require the "pre-calculation of the hull". Aside from divide and conquer the two most common methods are the sweep line approach (Fortune, 1978 and Edelsbrunner and Siedel 1986) and the incremental approach (Guibas *et al.* 1990).

Divide and conquer has also been successfully applied to develop linear time algorithms for diagram creation (Tsai 1993). Most divide and conquer algorithms approach the problem by first sorting or labeling the data points prior to the calculations. A grid is often developed to find the initial spatial point ordering (Maus, 1984). These grid based algorithms require extensive overhead to setup and maintain thus I developed the idea that if an algorithm could sort the points by using the hull geometry this would provide an effective means to pre-sort the data points. Linked to this division of the hull is the theory of maximizing topological order. The initial division of the hull into a triangular fan follows the theory but whether the subsequent triangulations resulted in a maximization of topological efficiency was unanswered. To prove the theory may well result in a design more effective than the current grid based methods. Also it is important to include the newer randomization approaches to the problem as these algorithms provide optimal lower bound solutions and are often unaffected by the distribution of the input datasets , unlike the deterministic ones mentioned previously.

Statistical tessellation provides a complex but rational way of predicting the conditional nature of spatial point patterns. However, I must caution a user of this technique to a fundamental spatial problem inherent to most spatial statistics, namely spatial autocorrelation. Spatial autocorrelation may play a role in the conditioning of wildlife to natural features. For example, if the behavior of moose was to reduce its contact with humans then the regular visits by humans to say oil well sites could prompt an avoidance of oil well sites by moose. In contrast, the moose may be associating with a feature because another highly favorable feature is present nearby. For example, riparian aquatic vegetation in a lake. It is also important to note that Rc values greater than one do not automatically mean that the wildlife are avoiding a certain feature. It could well be that the wildlife are attracted to another feature which is itself avoiding the initial landscape features. The isolation of features and a knowledge of the spatial associations will provide insight into the rational for the behavior observed in the statistical outcomes.

Overall, the non-biased nature of tessellation statistics provides a vigorous measure of the conditional dependency that any landscape feature may have on the distribution of wildlife, which in turn should help towards improving our understanding of the underlying wildlife behaviours that create these distribution patterns It is conceivable that within the framework of ecological theory tessellation may well contribute considerable knowledge to the many unsubstantiated theories already developed and it is hoped that it will lend credibility to new and exciting hypothesis concerning the spatial relationships between wildlife and their habitats.

5.1 Bibliography

Edelsbrunner, H. and R. Siedel. 1986. Voronoi diagrams and arrangements. Disc. Comp. Geom. 1,: 25-44.

Fortune, S. 1978. A sweepline algorithm for Voronoi diagrams. Algorithmica 2: 153-174.

Guibas, L.J., D.E. Knuth and M. Sharir. 1990. Randomized incremental construction of Delaunay and Voronoi diagrams Springer LNCS 443: 414-431.

Johnson, D.H. 1980. The comparison of usage and availability measurements for evaluating resource preference. Ecology 61: 65-71.

Maus, A. 1984. Delaunay triangulation and the convex hull of n points in expected linear time. BIT 24: 151-163.

Tsai, V.J.D. 1993. Fast topological construction of Delaunay triangulations and Voronoi diagrams. Comp. & Geosci. 19(10): 1463-1474.