UCGE Reports
Number 20226

Department of Geomatics Engineering

Algorithms for Automatic Vectorization of Scanned Maps

(URL: http://www.geomatics.ucalgary.ca/links/GradTheses.html)

by

Girija Dharmaraj

July 2005

THE UNIVERSITY OF CALGARY

Algorithms for Automatic Vectorization of Scanned Maps

by

Girija Dharmaraj

A THESIS
SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF GEOMATICS ENGINEERING

CALGARY, ALBERTA
JULY, 2005

Abstract

The proliferation of Geographic Information System (GIS) in industry and research has lead to the need for converting the available analog geospatial data to digital form. Although maps can be scanned, they cannot be used directly in a GIS system without processing. All the available commercial raster to vector conversion software are semi-automatic and require an operator for digitization and verification. Thus automatic algorithms are required for faster and reliable conversion of maps where the operator is required only for verification.

Methods for automatic vectorization of scanned maps deal with polygons, lines and points. The focus of this research is on the extraction of linear features from scanned maps and satellite imageries using skeletonization. Two methods that are examined are skeletonization by Voronoi Diagram and Mathematical Morphology. Connectivity is established between disconnected lines using Least Square Parabola (LSP). The objectives of this research are to compare the two vectorization methods and examine the application of LSP for gap filling.

Mathematical Morphology is a method for quantitative analysis of spatial structures that aims at analyzing shapes and forms of an object. Thinning is a morphological operation that is used to remove selected foreground pixels from binary images. A Skeleton is basically the centerline extracted from an object that results from thinning.

Let $\mathcal{O}$ be a set of sites in Euclidean space of dimension n. For each site o of $\mathcal{O}$, the Voronoi cell $V(o)$ of o is the set of points that are closer to o than to other sites of $\mathcal{O}$. The Voronoi diagram $V(\mathcal{O})$ is the space partition induced by Voronoi

cells [33]. In this method first edge detection is done on the image and every point on the edge is taken as a generator point and the Voronoi Diagrams are generated using incremental algorithm using Quad Edge Data Structure. Then the skeletons are extracted from the Voronoi Diagrams.

The scanned black and white, and grayscale maps are skeletonized using Mathematical Morphology and Voronoi Diagrams. The results produced by these two methods are compared based on the accuracy of the result by calculating the mean and standard deviation and its ability to handle curves and junctions.

The conclusion of this research states that vectorization should be implemented using a method that gives the best result and that best suits the needs of the user's GIS system. Skeletonization by Mathematical Morphology preserves the geometry of the object, forces the skeleton to be in the middle of the object and produces one pixel width skeleton and its accuracy is also upto 3.5 meters. The resulting image is then imported into ArcGIS or Grass to be converted to Vector data and building the topology. Skeletonization by Voronoi Diagram preserves the geometry of the object, on an average the skeleton is approximately in the center of the object and the resulting data is in vector form and the topology is also built using the Quad edge data structure and its accuracy is also upto 5 meters. Therefore the main difference between both the skeletonizaion methods is that there is an extra step of importing and building the topology in Skeletonization by Mathematical Morphology, but, it results in the centerline of the object.

Acknowledgements

A journey is easier when one travels as a team. I started this journey in May 2003, and since then a large number of people have been quite generous with their support. It gives me immense pleasure to express my deep and sincere gratitude here to all people who accompanied me for the past two years when this research was being performed.

I cannot imagine going through this journey without the support and assistance of my supervisor Dr. Darka Mioc. I am deeply obliged to her for her stimulating suggestions, guidance and encouragement throughout the course of my research and while writing my thesis. I would like to thank her and Dr. Francois Anton for providing the program that they coded for generating skeletons using Voronoi Diagrams. I would slo like to thank my co-supervisor Dr. Ayman Habib for providing data and giving some suggestions on my research.

I would like to thank all my friends for the interesting discussions I had with them. Special thanks are due to my friends, Qiaoping Zhang, Ojaswa Sharma, Valarmathy Meenakshisundaram, Matthew Reid and Seema Phalke for our discussions and conversations on programming techniques and science in general.

It would be an act of injustice if I do not express my gratefulness to Ankur Saxena for his constant encouragement, support and his help with Latex for writing this thesis.

Most of all, my deepest gratitude goes to my parents and my brother for their love, unquestioned support and encouragement. It was their wishes and inspiration that propelled me through difficult times. I owe it all to them.

Table of Contents

Abstract ii

Acknowledgements iv

Table of Contents v

1 Introduction 1
- 1.1 Motivation and Problem Statement . . . 3
- 1.2 Research Objectives . . . 7
- 1.3 Methodology . . . 7
  - 1.3.1 Skeletonization by Mathematical Morphology . . . 8
  - 1.3.2 Skeletonization by Voronoi Diagrams . . . 9
  - 1.3.3 Gap Filling . . . 9
  - 1.3.4 Comparison of both algorithms . . . 10
- 1.4 Thesis Organization . . . 10

2 Previous Research 13
- 2.1 Introduction . . . 13
- 2.2 Raster to Vector conversion: Why and How? . . . 14
  - 2.2.1 Raster Data . . . 14
  - 2.2.2 Vector Data . . . 16
  - 2.2.3 Advantages of vector data over raster data . . . 17
  - 2.2.4 Raster to Vector Conversion methods . . . 19
- 2.3 Analysis of commercial vectorization software . . . 21
  - 2.3.1 TracTrix (2000) . . . 22
  - 2.3.2 AlgoLab Raster to Vector Conversion Toolkit . . . 23
  - 2.3.3 Able Software R2V for Windows . . . 24
  - 2.3.4 VextraSoft . . . 25
  - 2.3.5 WinTopo Professional . . . 27
- 2.4 Automatic Raster to Vector Conversion Methods . . . 29
- 2.5 Mathematical Morphology . . . 31
  - 2.5.1 Structuring Element . . . 32
  - 2.5.2 Dilation . . . 33
  - 2.5.3 Erosion . . . 34
  - 2.5.4 Opening . . . 36
  - 2.5.5 Closing . . . 37
  - 2.5.6 Thinning . . . 38

2.5.7 Pruning . . . 39
2.5.8 Connectivity Number . . . 39
2.6 Grayscale Mathematical Morphology . . . 40
2.6.1 Grayscale Thinning . . . 41
2.7 The Voronoi Diagram . . . 44
2.7.1 Delaunay Triangulation . . . 45
2.7.2 Quad edge data structure . . . 46
2.7.3 Methods for generating Voronoi diagram . . . 51
2.8 Thresholding . . . 53
2.8.1 Global Thresholding . . . 55
2.8.2 Local Thresholding . . . 57
2.9 Summary . . . 60

3 Skeletonization **61**
3.1 Introduction . . . 61
3.2 Skeletonization by Mathematical Morphology . . . 62
3.2.1 Introduction . . . 62
3.2.2 Methodology . . . 63
3.2.3 Skeletonization of Grayscale Maps by Mathematical Morphology 69
3.2.4 Skeletonization of Grayscale Maps by Mathematical Morphology 70
3.3 Skeletonization by Voronoi Diagrams . . . 73
3.3.1 Introduction . . . 73
3.3.2 Methodology . . . 73
3.3.3 Skeletonization of black and white Maps by Voronoi Diagram 76
3.3.4 Skeletonization of Grayscale Maps by Voronoi Diagram . . . 78
3.4 Applications . . . 80
3.4.1 Linear feature extraction from satellite imageries . . . 84
3.5 Summary . . . 91

4 Gap Filling **92**
4.1 Introduction . . . 92
4.2 Least Square Parabola . . . 93
4.3 Methodology . . . 95
4.4 Results . . . 108
4.5 Conclusion . . . 111
4.6 Summary . . . 111

5 Comparison of Results **112**
5.1 Handling of curves and junctions . . . 112
5.2 Error Analysis . . . 115

5.3 Conclusion 123
5.4 Summary 126

6 Conclusion and Future Work **127**
6.1 Discussion and Conclusion 127
6.2 Research Contribution 129
6.3 Future work 130
6.3.1 Automatic vectorization of Color maps and Color satellite imageries 130

Appendices **131**

A Commercial Vectorization Software **131**

Bibliography **141**

List of Figures

1.1 Flow Chart of Thesis 12

2.1 Example of Raster and Vector Data: a) Raster Data, b) Vector Data 15
2.2 Example of Raster and Vector Data (Zoomed): a) Raster Data, b) Vector Data 16
2.3 Example of Raster and Vector Data: a) Raster Data, b) Vector Data 17
2.4 Tractrix: a) Original image, b) Result of Vectorization 23
2.5 Algolab: a) Original Image, b) Result of Vectorization 24
2.6 Able: a) Original Image, b) Result of Vectorization 26
2.7 Vextrasoft: a) Original Image, b) Result of Vectorization 27
2.8 Wintopo Professional: a) Original Image, b) Result of Vectorization . 28
2.9 Comparision of Vectorization methods (source: Wenyin, L. and Dori, D. (1999) [63]) 30
2.10 Structuring Elements: a) Right-neighbor Correlation, b) 5 Cross, c) 3 X 3 33
2.11 Binary Dilation: a) Original Image, b) Structuring Element, c) Dilation 33
2.12 Scanned Black and White Map 34
2.13 Binary Dilation 35
2.14 Binary Erosion: a) Original Image, b) Structuring Element, c) Erosion 35
2.15 Binary Erosion 36
2.16 Binary Opening 37
2.17 Binary Closing 38
2.18 Pruning: a) Original Image, b) Thinning, c)Pruning 39
2.19 An Illustration of the connectivity number, a) The center pixel does not connect any region so it can be deleted. Connectivity number = 1, b) The center pixel cannot be deleted because the left and the right half would get disconnected. Connectivity number = 2, c) Connectivity number = 3, d) Connectivity number = 4 (maximum), e) Connectivity number = 0 40
2.20 Original Image with Three Intensity Values 42
2.21 Grayscale thinning: (a), (b) and (c) are binary images with intensity values of 255, 143 and 204 respectively and (d), (e) and (f) are the corresponding skeletons 43
2.22 Addition of skeletons 43
2.23 Template for Detecting a One-pixel-wide Ridge 44

2.24 Grayscale thinning: (a), (b), (c) and (d) are skeletons generated by convolving the original image with template T1, T2, T3 and T4 respectively . . . 44
2.25 a) Delaunay Triangulation, b) Voronoi Diagram . . . 45
2.26 Edge function (Source: [26]) . . . 47
2.27 a) Voronoi diagram (thick line) of points 1, 2 and 3, b) Corresponding Quad edge data structure (source: [34]) . . . 48
2.28 a) Voronoi diagram and Delaunay Triangulation, b) Corresponding Quad Edge Data structure (source: [34]) . . . 49
2.29 Voronoi Diagram of a set of generator points (P_1, P_2, P_3, P_4 and P_5) . . . 52
2.30 Addition of a new generator point P (Source: adopted from [39]) . . . 52
2.31 Final result of adding a new generator point . . . 53
2.32 Scanned Grayscale Map . . . 57
2.33 Thresholded Image . . . 58
2.34 Satellite Imagery . . . 58
2.35 Global Thresholding using Otsu's algorithm . . . 59

3.1 Classic thinning artifacts, a) Necking, b) Tailing, c) Spurious projection (source: [42]) . . . 64
3.2 Templates used for acute angle emphasis preprocessing step (source: [42]) . . . 65
3.3 a) Pixel marked for deletion in 1st iteration, b) Pixel marked for deletion in 2nd iteration . . . 66
3.4 Holts Staircase Problem. The center pixel in one of the above can be deleted . . . 68
3.5 Experimental Results, a) Original Image, b) Skeleton . . . 69
3.6 Scanned Map (Source: Library, University of Calgary) . . . 69
3.7 Skeletonization by Mathematical Morphology . . . 70
3.8 Grayscale map . . . 71
3.9 Skeletonization by Mathematical Morphology (Grayscale scanned map) 71
3.10 Grayscale thinning (convolved with template 1) . . . 72
3.11 Grayscale thinning (convolved with template 3) . . . 72
3.12 Original Image . . . 76
3.13 Voronoi Diagrams . . . 77
3.14 Delaunay Triangulation . . . 77
3.15 Skeleton using Voronoi Diagram . . . 78
3.16 Thresholded Grayscale Map . . . 79
3.17 Voronoi Diagram of Thresholded Grayscale Map . . . 80
3.18 Delaunay Triangulation of Thresholded Grayscale Map . . . 81
3.19 Skeleton using Voronoi Diagram . . . 81

3.20 Voronoi Diagram of Grayscale Map 82
3.21 Delaunay Triangulation of Grayscale Map 82
3.22 Skeleton and Crust of Grayscale Map 83
3.23 Flowchart of methodology for skeletonizing satellite image 85
3.24 Original Satellite Image (Source: Dr. Habib) 86
3.25 Grayscale Opening 86
3.26 Preprocessed binary image 87
3.27 Skeletonization by Mathematical Morphology 88
3.28 Skeletonization by Voronoi Diagrams 88
3.29 Satellite Image 89
3.30 Skeletonization by Mathematical Morphology 89
3.31 Skeletonization by Voronoi Diagrams 90
3.32 Satellite image 90
3.33 Skeletonization by Mathematical Morphology 90
3.34 Skeletonization by Voronoi Diagrams 91

4.1 Flowchart for Gap Filling 95
4.2 Case 1: Gap Filling 99
4.3 Example of Gapfilling (case 1) 100
4.4 Case 2: Gap Filling 101
4.5 Example of Gapfilling (case 2) 101
4.6 Case 3: Gap Filling 102
4.7 Example of Gapfilling (case 3) 103
4.8 Case 4: Gap Filling 103
4.9 Example of Gapfilling (case 4) 104
4.10 Case 5: Gapfilling 105
4.11 Example of Gapfilling (case 5) 106
4.12 Case 6: Gapfilling 107
4.13 Example of Gapfilling (case 6) 108
4.14 Original Image 109
4.15 Gap Filled Image 109
4.16 Contour Map with Gap 109
4.17 Gap Filled Contour Map 110

5.1 Original Image 113
5.2 Skeletonization by Mathematical Morphology 113
5.3 Skeletonization by Voronoi Diagrams 114
5.4 Comparison of Curves 114
5.5 Comparison of Junction 115
5.6 Comparison of corners 115

5.7 Scanned Map 1 117
5.8 Manually Digitized (Scanned Map1) 117
5.9 Skeletonization by Mathematical Morphology 118
5.10 Skeletonization by Voronoi Diagrams 118
5.11 Scanned Map 2 119
5.12 Manually Digitized (Scanned Map2) 119
5.13 Skeletonization by Mathematical Morphology 120
5.14 Skeletonization by Voronoi Diagrams 120
5.15 Overlay of Skeltonization by Mathematical Morphology on Scanned map 1 121
5.16 Mean, Standard deviation and RMSE of Map 1 124
5.17 Mean, Standard deviation and RMSE of Map 2 125

Chapter 1

Introduction

"If a picture is worth a thousand words, a map is worth a thousand pictures"

- Christel Binnie

Over the past 50 years the pace of technological advancement has far exceeded the impact of industrial revolution. Paper maps have always been of prime importance in the field of surveying. Technological advancement has led to the introduction of the Geographic Information System (GIS) in the late 1960's and early 1970's, which requires digital maps for processing. GIS is defined by Gottfried Konecny (2003) [29] as,

"A computer system for the input, manipulation, storage and output of digital spatial data. It is a digital system for the acquisition, management, analysis and visualization of spatial data for the purpose of planning, administering and monitoring the natural and socio-economic environment."

Thus a GIS is a computer system that is capable of acquiring, storing, integrating and displaying spatially organized information [2] and [30]. It is the arrangement of computer hardware, software, and geographic data which is used to integrate, analyze, and visualize the data. It is also used to identify relationships, patterns, and trends. A GIS is typically used to represent maps as data layers that can be studied and used to perform analysis [9].

In todays society, there is a greater emphasis on digital data than on paper maps. This corresponds to the proliferation of GIS in industrial and research environment. This has lead to the the need to transform preexisting paper maps to digital maps. For example, if we take into consideration a historical paper map, all the required information about the map is available only on that map. If someone wants to access this map from a different location then this map can be sent over to that location or it could be scanned and a digital copy can be sent to that location. But, if this map is digitized into different layers and all the required information about the map is added as attributes then it becomes very easy to send and do any kind of spatial analysis on that map. It becomes easy to store and transfer the data since the size of vector data is smaller than that of raster data. It also becomes easy to update and manage the data. Thus, there is a need to convert all the available geospatial data in paper form to digital data, which resulted in the need for **Raster to Vector conversion of Maps, or Vectorization**.

Raster data can be defined as an abstraction of the real world where spatial data is expressed as a matrix of cells or pixels, with spatial position implicit in the ordering of the pixels. A raster map is a map stored as a regular array of cells [32]. In GIS Vector data can be defined as positional data in the form of points, lines and polygons, expressed in x and y coordinates [32]. These vector data have topological relationship, location and attribute information associated with them. Attributes are other information that is used to describe the map information that is represented by a point, line, or polygon. For example, an attribute of a polygon might identify it to be a lake or swamp; an attribute of a line might identify it to be a road, railroad, stream, or shoreline.

Vectorization involves digitization of the paper maps and adding attributes. In this thesis we will focus only on automatic digitization since, automatically adding attribute information like street names and building names is not possible.

The advantages of automatic digitization are [10]:

1. The scale of the map or drawing can be changed i.e. the scale of the map can be increased or decreased to any desired limit and the area of any closed polygon can be calculated.

2. Different layers of the map can be created and different layers can be switched on or switched off to view only the desired layers which make the map more readable and less complex.

3. The width of the lines, arc, circles etc., of the map can be changed and a 3D model can be created. The vector data can be viewed and analyzed in 2D/3D which can help in fast and low cost decision making.

4. Updating the map becomes very easy. New maps can be created by extracting some part of the original map and merging it with some other part of another map.

The next sections gives a description of the motivation and the problem statement based on the advantages of vectorization given above.

1.1 Motivation and Problem Statement

Digitization can be done manually, semi automatically or automatically. Manual digitization is done by using a digitizer board or by online digitization. Online

digitization is done by scanning the map and digitizing on the computer screen using the mouse. Manual digitization is time consuming and can generate errors that may result in inaccurate maps. These maps might also require a considerable amount of processing time in order to build the topology. For example, while digitizing a map different features have different width (thick or thin lines) with respect to their importance. The digitizer has to digitize the thick lines through the center which is error prone and not easy to do manually. However in manual digitization gaps between lines can be filled by hand. Semi automatic digitization involves some level of interaction between the user and the computer. Automatic digitization is done by using different techniques/algorithms for center line extraction. The map is scanned and an algorithm is run on it which results in a digitized map. The process is fast (few seconds) and it has good precision.

In most developed countries, the collection and use of vectorized data is well established. Due to the availability of funding, new techniques, such as mobile mapping, are constantly being developed and applied. There is also a need for vectorized data in developing and under developed countries. However, they often do not have access to the same techniques applied in developed countries mainly due to a lack of funding. This research can help digitize paper maps in countries or regions (state/province of developed countries) that can afford to digitize their preexisting paper maps but not survey again for data acquisition using new techniques.

In todays GIS, vector data models have great importance. Various vendors like ESRI, Intergraph, Siemens and many others have developed GIS software packages that use vector data models. Some of the common modules that they provide are [29]:

1. Map transformation (translation, rotation and scaling)
2. Edge matching (fit lines of adjacent lines together)
3. Intersection between layers (union, intersection, etc.)
4. Buffer zone generation of point, line and polygons
5. Measurement of point coordinates, length and area
6. Interpolation
7. Modelling function
8. Symbol and text generation
9. Generalization
10. Map annotation

Performing spatial analysis like buffer, intersection etc. on vector data is fast, easy and efficient. Hence there is a need for automatic rater to vector conversion of maps.

Automatic Vectorization is not possible because we can only automate the process of digitizing but not adding the attributes. Presently, there is no complete automatic digitization software that exists in the commercial market. The list of semiautomatic software available in the commercial market is given in the appendix (A) Commercial Vectorization Software). A comparison of some of the commercial vectorization software is shown in Chapter 2. There are many aspects involved with automatic digitization of scanned maps. There are methods that deal with polygons, linear

features and points. The focus of this research is on the extraction of centerlines of the objects from scanned maps. Some of the methods that are used for digitization of linear features are: Thinning-based methods, Contour-based methods, Run-graph-based methods, Mesh-pattern-based methods and Sparse-pixel-based methods [63]. A description of the mentioned methods is given in Chapter 2. The medial axis transform is useful in thinning a line and thinning results in the skeleton [49]. Two algorithms that are examined and compared are skeletonization by Mathematical Morphology and Voronoi Diagrams.

Mathematical Morphology is a method for quantitative analysis of spatial structures that aims at analyzing shapes and forms of an object. A Skeleton is basically the centerline extraction of an object that results from thinning. In thinning, a structuring element (kernel) is passed over every pixel in the image and is compared with the underlying image pixels. If the foreground and background pixels in the structuring element exactly match the foreground and background pixels in the image, then the image pixel underneath the origin of the structuring element is set to that of the background otherwise it is left unchanged.

Let $\mathcal{O}$ be a set of sites in Euclidean space of dimension n. For each site o of $\mathcal{O}$, the Voronoi cell $V(o)$ of o is the set of points that are closer to o than to other sites of $\mathcal{O}$. The Voronoi diagram $V(\mathcal{O})$ is the space partition induced by Voronoi cells [33]. In this method, first edge detection is done on the image and every point on the edge is taken as a generator point and the skeleton is generated.

1.2 Research Objectives

The primary objective of this research is to extract the centerline of the lines on a black and white, and grayscale scanned map using skeletonization by Voronoi Diagrams and Mathematical Morphology and compare the result obtained by using both the algorithms. Another primary objective is to fill gaps between the skeletonized lines.

The secondary objective is to suggest a methodology for linear feature extraction of satellite imageries using skeletonization and review the existing algorithms and the commercial software available for automatic vectorization of scanned maps.

The next section gives a brief description of the methodology.

1.3 Methodology

The methodology is based on image processing algorithms for the extraction of spatial feature like line drawings from raster data. The basic steps for any automatic vectorization involve [45]:

1. Digitization of the original paper document using a scanner
2. Filtering
3. Thresholding
4. Thinning and pruning the binary image
5. Gap filling
6. Raster to vector conversion.

Different algorithms could be used for the first four steps to obtain good results but, on the contrary, the fifth step gives different results depending on the applied method [45]. The next section briefly describes the methodology used for Vectorization by Mathematical Morphology.

1.3.1 Skeletonization by Mathematical Morphology

Mathematical Morphology is a method for quantitative analysis of spatial structures that aims at analyzing shapes and forms of an object. A Skeleton is basically the centerline extraction of an object that results from thinning.

Vectorization of a black and white map by Mathematical Morphology using skeltonization is done by thinning. Thinning is the process of reducing a shape to a simpler version that still retains the essential features of the original object. The thinned version of the shape is called the skeleton (centerline). When the skeleton is centered with respect to the boundaries of the original object, it is referred to as the medial-axis or medial-surface [49], [42] and [47]. Thinning can be applied only to binary images. In this research Stentifords algorithm is chosen and used as a preprocessing step to remove classical thinning artifacts. Then the resultant image is thinned using Zhang-Suen's algorithm because this method is used as a basis for comparison of thinning methods for many years [42]. Finally the resultant image is post processed to remove step like structures using Holt's algorithm [42]. Finally the thinned binary image is vectorized using GRASS or ArcGIS. The next step is to vectroize grayscale maps.

Grayscale maps can be vectorized in two ways. The map can be converted to a binary image by segmenting or thresholding and the combination of the above men-

tioned algorithm can be applied or the grayscale map could be directly skeletonized using Weiss's algorithm [60] and [61]. The next subsection briefly describes how vectorization of scanned map is done using Voronoi Diagrams.

1.3.2 Skeletonization by Voronoi Diagrams

The approach is based on image processing techniques to extract the basic spatial features from raster data. In our method first edge detection is done on the black and white, or grayscale image and every point on the edge is sampled as a generator point. Then Voronoi Diagrams are generated using the incremental method using the quad edge data structure. Then the skeleton is extracted from the Voronoi Diagram.

The next subsection shows how disconnected lines can be connected using a gap filling algorithm.

1.3.3 Gap Filling

After skeletonization and pruning the most important task is to reconnect the line that are disconnected by other information layers like symbols, grids etc [5]. We first skeletonize the lines and then try to find relevant extremities of the skeleton for recovering the missing parts of the lines. Connectivity is established between disconnected lines using Least Square Parabola (LSP) fitting. A detailed description of LSP and its implementation for establishing connectivity (gap filling) between disconnected lines is shown in Chapter 4.

1.3.4 Comparison of both algorithms

The algorithm of vectorization by Voronoi Diagrams and Mathematical Morphology are compared based on the following criteria:

- How does it handle curves and junctions?
- Error Analysis

The mean and standard deviation are calculated to compare the accuracy of skeletonization by both the methods.

1.4 Thesis Organization

In order to meet the objectives defined in section 1.3, a systematic approach to analysis, implementation and comparison of the algorithms has been adopted. In this Chapter we have seen the motivation and problem statement of this research, i.e. why this research is important in the raster vector conversion in GIS. A small summary of all the chapters are given below.

Chapter 2, consists of reviewing and evaluating the existing spatial data applications for raster/vector conversion in GIS. This chapter starts with the definition of raster and vector data structures and then proceeds with some of the questions like why raster to vector conversion is required and how it has been done till today. Then it gives a brief description of topology and its importance in GIS. It also gives a brief idea of the vectorization software available today in the commercial world and compares them with respect to the modules it provides and the quality of the output it produces.

This chapter shows how and why thinning (skeletonization) by Mathematical Morphology and Voronoi Diagrams was chosen for automatic digitization. Then, it gives a description of previous research in thinning using Mathematical Morphology and Voronoi Diagram. After that, it shows how the centerline can be extracted from scanned maps using skeletonization by Mathematical Morphology and Voronoi diagram. The description and the usefulness of Voronoi Diagram, Delaunay Triangulation, Quad edge data structure, Mathematical Morphology etc. with respect to the research perspective is shown in detail.

Chapter 3 shows the methodology adopted to automatically digitize scanned maps using skeletonization by Mathematical Morphology and Voronoi Diagrams. It also gives a brief description of the applications of this research and gives the methodology of one such application which is linear feature extraction from satellite imageries.

Chapter 4 shows how connectivity can be established between disconnected lines using Least Square Parabola (LSP). A detailed description of LSP and its implementation for establishing connectivity (gap filling) between disconnected lines is shown in this chapter.

Chapter 5 shows the results and compares the vectorized map with respect to the way it handles curves and junctions and error analysis by calculating the mean and standard deviation.

Chapter 6 gives the conclusion and future work. The conclusion describes the motivation and objectives of this research. Then the main research contributions are presented and finally, some recommendations for further research are made.

The flowchart of the thesis organization is shown in the figure 1.1

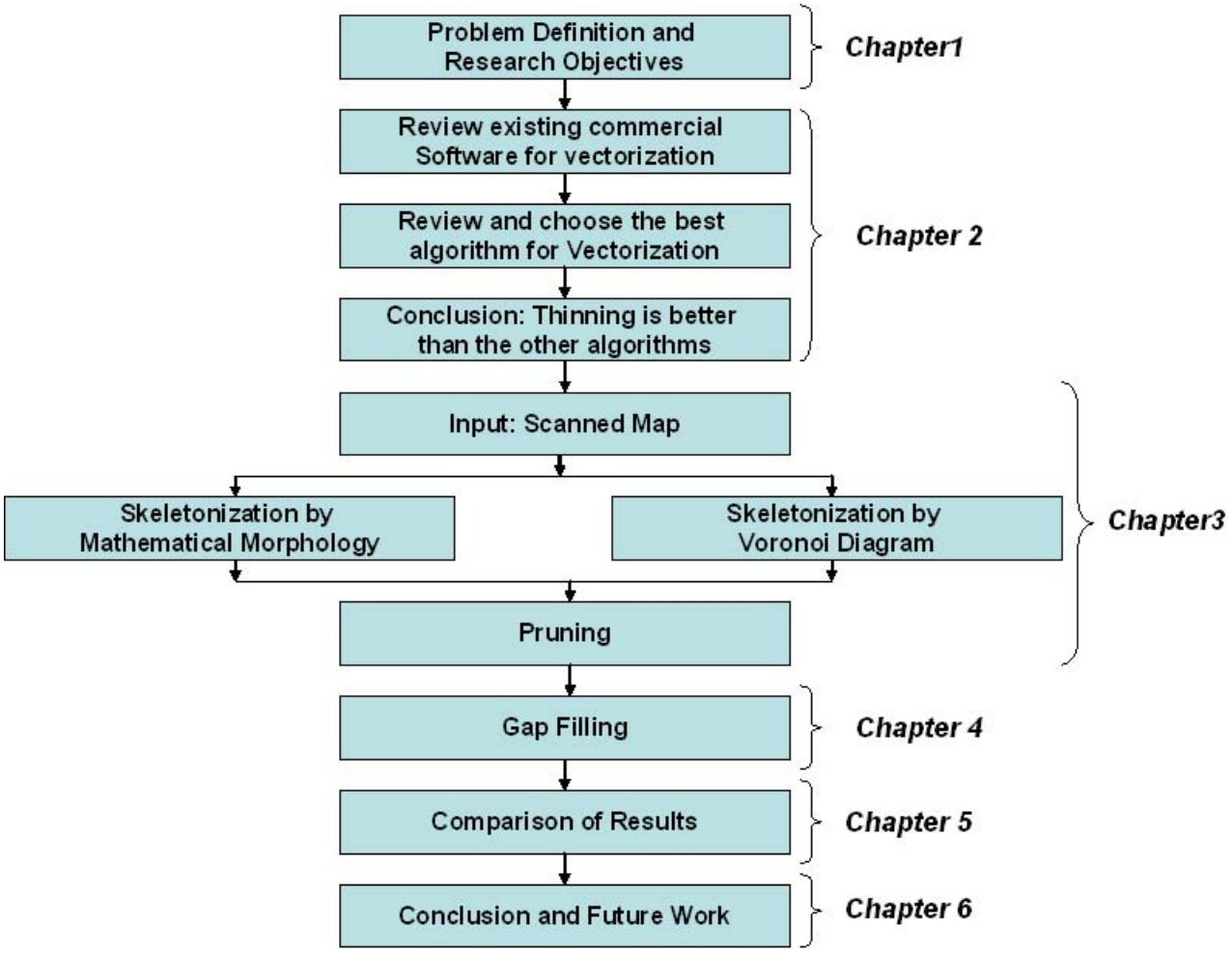

Figure 1.1: Flow Chart of Thesis

Chapter 2

Previous Research

2.1 Introduction

The two basic data structures that are used for storing and manipulating images and graphics on a computer are raster and vector data models. All of the major GIS (Geographic Information Systems) and CAD (Computer Aided Design) software packages available today are mainly based on one of the two data structures, either raster or vector [10]. This chapter is divided into seven sections. The first section starts with the definition of raster and vector data structures and gives the advantages of vector data structures on raster images. It also explains the need for raster to vector conversion of maps.

The second section gives a brief description of the different commercial vectorization softwares available in the market. It shows their advantages and disadvantages. The result of vectorization using these software on a scanned map are also shown. A list of some of the commercial vectorization softwares and a brief description of them are given in the appendix (Chapter A) Commercial Vectorization Software).

Image processing techniques were introduced more than 30 years ago and many basic vectorization methods have been developed and implemented since then. These methods can be approximately divided into six classes [63] and they are, Hough Transform based methods, Thinning-based methods, Contour-based methods, Run-graph-based methods, Mesh pattern-based methods, Sparse-pixel-based methods and

Line tracing methods [63].

Thinning is an act of identifying a skeleton [42]. A map can be thinned or skeletonized using two methods, viz. Mathematical Morphology and Voronoi Diagrams. Section 2.5 - "Mathematical Morphology", gives the definition of Mathematical Morphology and some Morphological operations. It also shows the properties of the different morphological operations used in this research. Section 2.6 gives a brief description of grayscale morphology.

Section 2.7 - "Voronoi Diagrams", starts with the definition of Voronoi Diagram, Delaunay triangulation etc. and describes the quad edge data structure. Then it gives a brief explanation of the three different methodologies used to construct the Voronoi diagrams, Incremental method, Plane Sweep Method and Divide-and-Conquer Method [39].

Section 2.8 - "Thresholding", shows the importance of thresholding in image processing and gives an explanation of the different thresholding methods. It also shows the use of histograms in that process. Otsu's [40] method for global thresholding is also described in this section.

2.2 Raster to Vector conversion: Why and How?

2.2.1 Raster Data

Raster data can be defined as an abstraction of the real world where spatial data is expressed as a matrix of cells or pixels, with spatial position implicit in the ordering of the pixels. The pixel value indicates the attribute, such as color, elevation, or ID number [32]. In figures 2.1, 2.2 and 2.3, (a) is raster data and (b) is vector data.

Figure 2.1: Example of Raster and Vector Data: a) Raster Data, b) Vector Data

Raster images are usually acquired by optical scanners, digital CCD cameras or other raster imaging devices. It is represented by a two-dimensional array of pixels. Its spatial resolution is determined by the resolution of the acquisition device (quality and resolution of the camera, dpi used for scanning the paper map) and the quality of the original data source (small scale, large scale map). The geometric accuracy of raster data is limited by the cell resolution [29]. Since the raster image is made of pixels, the spatial locations of the objects are represented by its spatial resolution. If the spatial resolution is increased by 2 times then the total size of a two-dimensional raster image will increase by 4 times because the number of pixels is doubled in both X and Y dimensions [29]. In such cases the image becomes more and more rough, "notched"or "jagged" while zooming in. The example of a zoomed image is shown in figure 2.2.

One of the shortcomings of working with a raster image is that we can easily draw a line or a circle using a raster editor program like Microsoft Paint (Windows) or XV

(Macintosh), but if we try to change the color, position, size, or simply delete them then it becomes very tedious and we'll have to erase or apply changes to every pixel. These are some of the disadvantages of raster image because there is no explicit description about the geometry and size of the object. That is the main reason why the programs, which use computer graphics for calculations like, CAD/CAM, GIS, animation programs etc. uses another form of graphic data presentation, i.e. vector images (Vector Data) [10].

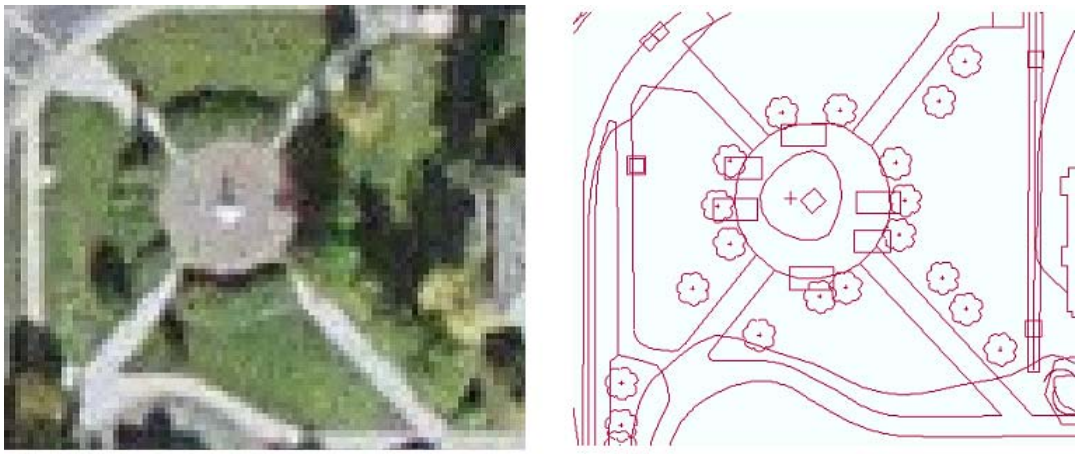

Figure 2.2: Example of Raster and Vector Data (Zoomed): a) Raster Data, b) Vector Data

2.2.2 Vector Data

Vector data, when used in the context of spatial or map information, refers to a format where map information is stored as points (node), lines (polylines) or polygons (areas). Vector data are geometrically and mathematically associated [32]. Points are stored using the coordinates, for example, a two-dimensional point is stored as (x, y). Lines are stored as a series of point pairs, where each pair represents a straight line

segment, for example, (x_1, y_1) and (x_2, y_2) indicates a line from (x_1, y_1) to (x_2, y_2). A line string is defined by the coordinates of all points forming the line string: $(x_1, y_1), (x_2, y_2), ...(x_n, y_n)$ [29]. Vector data has topological relationship, location and attribute information associated with it. Attributes are used to describe the map information represented by a point, line, or polygon. For example, an attribute of a polygon might identify as a lake or swamp; an attribute of a line might identify it as a road, railroad, stream, or shoreline. An example of a vector data is shown in figure 2.1 b, 2.2 b and figure 2.3 b.

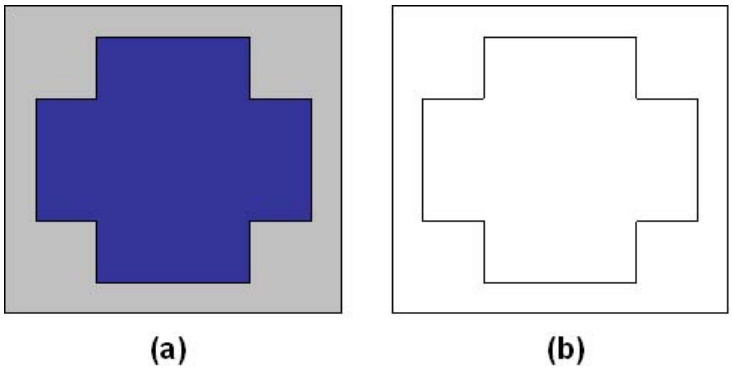

Figure 2.3: Example of Raster and Vector Data: a) Raster Data, b) Vector Data

2.2.3 Advantages of vector data over raster data

The advantages of vector data over raster data are as follows [10]:

1. The file size of vector data structures are smaller than those of raster images because a raster image needs to store the digital number of all the pixels while only point coordinates are stored in vector data. For example, let us consider an image consisting of large homogeneous regions as shown in the figure 2.3. In this example, the size of the raster image is 136 KB and corresponding

vector data is 200 bytes. Since we are concerned only about the boundaries and shapes of the regions (objects) storing the file as a vector is a good option.

2. Vector data structure is also a good option when geometric shapes need to be represented precisely in a GIS or Computer Aided Design (CAD) system because it is not limited by spatial resolution or pixel size. Mathematical transformation can be easily applied to regular shapes and smooth curves.

3. Most important of all vector data has topology which is very important in a GIS system. Topology is the relative location of a spatial feature that is independent of its exact position. Topological relationships such as connectivity, adjacency and relative position are usually expressed as relationships between points, lines and polygons [32] and [9]. The knowledge of topological relations permits neighborhood queries [29]. For example, vector data structure provides an easy description of the regions on the left or right side of a common boundary or if a point is in or out of a polygon area.

4. Another main advantage of vector data structure is flexible data manipulation that it has over raster images. Vector data is flexible in that it can be resized without losing resolution, it can be scaled, a portion of the map can be selected etc. For example, graphical features such as rivers and roads in a map viewed with a real-world projection system can be easily displayed at any scale without physically changing the data. The next three points demonstrates the idea of flexibility of vector over raster data.

5. The scale of the map or drawing can be easily adjusted i.e. the scale of the

map can be increased or decreased to any desired limit and the area of any closed polygon can be calculated.

6. Different layers of the map can be created. Layers can be switched on or off to view only the layers of interest are shown making the map more readable and less complex.

7. The width of the lines, arc, circles etc., of the map can be changed and a 3D model can also be created. Vector data can be viewed and analyzed in 2D/3D which can help improve the efficiency and cost effectiveness of decision making.

8. Vector data can be easily updated. New maps can be created by extracting portions of the original map and merging it with some other part of another map.

The next section gives a detailed description of raster to vector conversion methods.

2.2.4 Raster to Vector Conversion methods

Vector data is normally created from existing paper maps or natural source images, such as aerial photographs or satellite imagery. Vector data has traditionally been acquired by manual digitization. Manual digitization can be split into two categories, table top digitizing and overlay digitizing.

Table top digitizing is a fully manual method. In this method an operator uses a digitizing table with a hard copy map affixed to its surface. The operator registers its co-ordinates and proceeds to trace the map using a digitizing cursor which interacts

with the tables surface. Using this information a representation of the map in vector form is generated. The disadvantage of manual digitizing is that it can be time consuming and it also can be less accurate. The main reason being that a human eye has the capacity of distinguishing only 40 dots per inch (DPI) [10]. For a typical contour map, it can take one skilled operator approximately one or more weeks to trace all the lines manually [10]. It also requires a lot of time for manual error checking. The intensive labor requirement makes large mapping and GIS project difficult and expensive to implement.

Overlay digitizing was seen as a temporary solution. It was developed when many GIS systems began to support raster images as back drops. This allowed digitizing on-screen by marking out elements using the screen cursor. One of the advantage of this method was that the operator could zoom and pan the image to distinguish between lines or areas. Most of the semiautomatic techniques uses line tracing algorithms. The user adds vertices where the direction of the line changes.

Fully Automated Techniques that convert raster-to-vector were assumed to be the next step in GIS data capture. However, the expectation for a definitive solution faded in the face of the richness and complexity of geographic and engineering information held on maps and plans. Nearly all GIS applications require some of their basic data to be in vector form [10]. An improved solution was necessary to provide better and more efficient solutions to manual digitizing. Thus, the interactive semi automated approach was introduced.

The next section shows the advantages and disadvantages of vectorizing a scanned map using commercial software. The resulting vector map is also shown. All these softwares are based on interactive semi automated approach.

2.3 Analysis of commercial vectorization software

Line tracking and interactive vectorisation is one of the automated digitization techniques used currently. These methods involve some level of interaction between the user and the computer. Until today, there is no complete automatic vectorization software available. One of the reasons being that every image is different and it has its own properties. There are approximately 20 vectorization software packages available in todays commercial world (Google Search Engine, 2003). The name of the software, its website and a small description as given by the respective companies are given in the appendix (Chapter A) Commercial Vectorization Software).

Due to time constraints and taking into consideration the availability of the software, and system requirements to run the software the demo versions of some of these software were downloaded and tested for a portion of a scanned map. The reason for using a small portion is that the demo version of some softwares allowed only image sizes of 255 x 255. Some companies did not provide the demo version but they had a facility through which the customer sends by attaching a sample scanned map. They process this map and send back the result of vectorization using their software. Unfortunately, they provided this facility for commercial clients only. The algorithms used to develop the software were not made public or known by the companies. Hence, the software's are analyzed based on the output it generates and some other factors like the extent of user interaction required. The advantages and disadvantages of the softwares are given. The result of vectorization is also shown.

2.3.1 TracTrix (2000)

Advantages [28]

1. Centerline or outline can be produced.

2. In an image the text can be blocked (erased) manually using a raster editor and only the lines can be vectorized.

3. If the drawing contains different colors then these can be separated on different layers by selecting "Color separation", using color histogram.

4. Maximum line width for creating centerlines can be given which can remove noise and if the width of the line is too small (< 3pixels) then the noise level can be set to zero (everything will be vectorized).

5. Text TracTrix 2000 provides the user with the ability to create and save custom libraries of specific fonts. For example, you might create a specific library to be used when converting drawings hand-drawn by one or more individuals. This library would be trained to recognize the characters created by them.

Disadvantages [28]

1. The input image has to be cleaned and broken lines have to be connected before vectorizing. The raster image has to be preprocessed manually to clean imperfections using raster editing tools before vectorization.

2. Text TracTrix 2000 can recognize fonts but the user has to train the library before using it like in most of the Optical Character Recognition software.

The result obtained by vectorizing the scanned map using Tractrix is shown in figure 2.4.

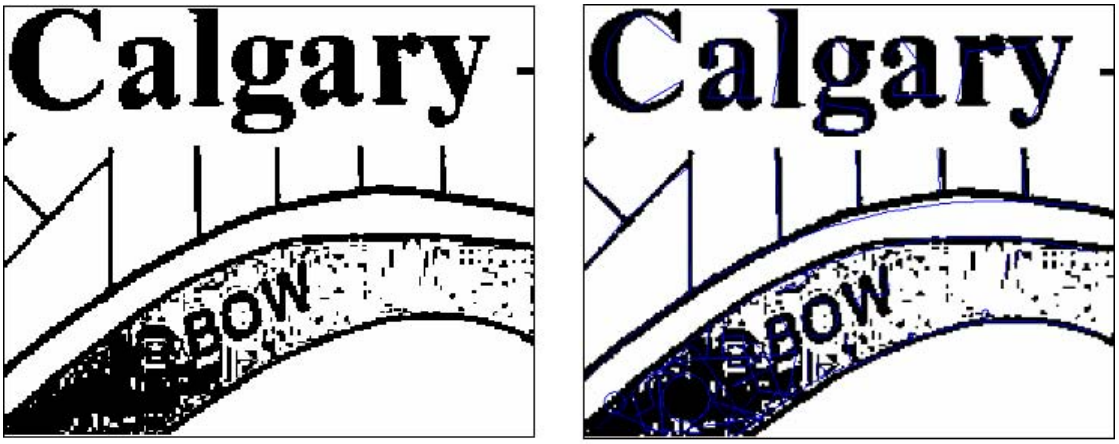

Figure 2.4: Tractrix: a) Original image, b) Result of Vectorization

The result was not satisfactory due to the dithering effect and the lines were jagged and discontinuous. Thus while developing an automatic vectorization algorithm the image must be filtered to remove the dithering effect.

2.3.2 AlgoLab Raster to Vector Conversion Toolkit

Advantages [1]

1. Centerline or outline can be produced
2. Curves the line
3. Thickness of the line and size of the image is not a problem
4. Versions of the s/w below 2.0 are free of charge

Disadvantages [1]

1. Preprocessing of the raster image is very important to produce a clean vector output which can be done using raster function (built in the software)
2. The lines are curved if they have small sharp edges
3. It cannot recognize text
4. Some parts of the line or curves are not vectorized
5. This s/w works only on binary image

The result obtained by vectorizing the scanned map using AlgoLab, Raster to Vector Conversion Toolkit is shown in figure 2.5.

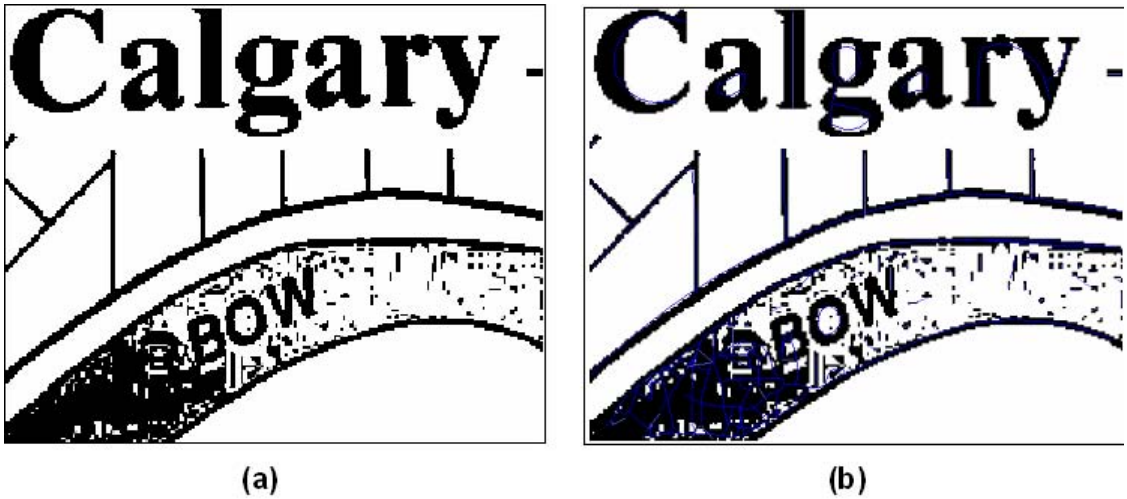

Figure 2.5: Algolab: a) Original Image, b) Result of Vectorization

2.3.3 Able Software R2V for Windows

Advantages [10]

1. Centerline or outline can be produced

2. The raster to vector (R2V) for Windows software is written in the Microsoft Windows environment. It has a graphical user interface that includes icons, menu bar, floating pop up menu and interactive on-line helps

3. Learn R2V in 30 min tutorial is available in help

4. Vector lines can be edited

5. The user can define as many layers as he needs and define properties like name, color etc. Layers can be created and modified using Edit/Layer Define function. Layers can also be copied or moved between layers

6. It uses OCR for text editing

Disadvantages [10]

1. This software works only on black and white images after noise removal

2. When converted to shape files, the image inverts (upside down mirror image is produced) due to the bug in the software

3. Grayscale image has to be converted to black and white and the noise has to be removed and color image has to be converted to grayscale and then processed

The result obtained by vectorizing the scanned map using Able Software, R2V for Windowst is shown in figure 2.6.

2.3.4 VextraSoft

Advantages [58]

1. Centerline or outline can be produced

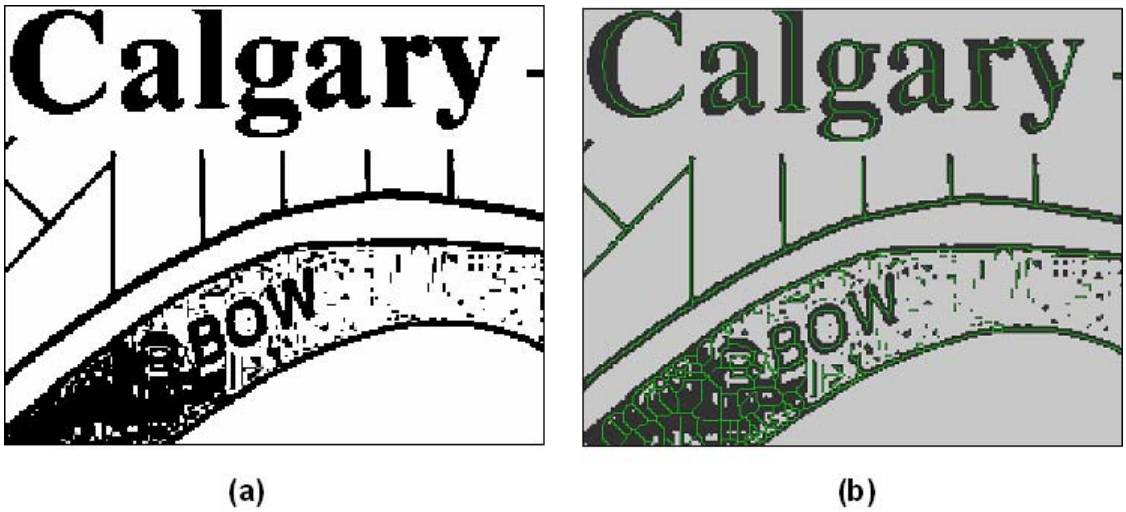

Figure 2.6: Able: a) Original Image, b) Result of Vectorization

2. It recognizes arcs, orthogonal lines and circles

3. It has the ability to perform, Polyline smoothing, Vector editor, Noise filtration and Input Z-values (elevation)

4. They have split the window into 3 windows which are: Main view (contains image at 1:1 scale in raster coordinates), Overview (always shows the whole image) and Microscope view (shows selected parts with fixed zoom)

5. The map can also be georeferenced

Disadvantages [58]

1. This software works only on black and white images after noise removal, so raster editing is very important

2. In their manual they state that there is an option to Convert grayscale scanned maps to black and white maps. But it does not exist in the demo version

3. This s/w uses spline so the lines are curved and disconnected. The centerline is not given

4. Text editor is not provided by this software

The result obtained by vectorizing the scanned map using VextraSoft is shown in figure 2.7.

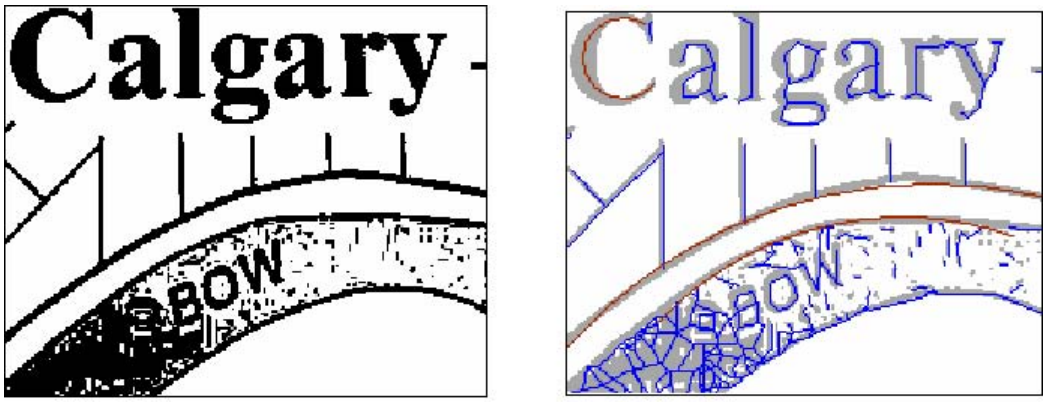

Figure 2.7: Vextrasoft: a) Original Image, b) Result of Vectorization

2.3.5 WinTopo Professional

Advantages [31]

1. WinTopo Pro vectorises colour images into colour CAD drawings

2. It has the ability to recognize arcs

3. Smooth arcs are fitted to the curved parts of the image

4. Some GIS analysis like georeferencing and translation can be done

5. It also has modules for Digitizing on board, vector Editing and raster editing

Disadvantages [31]

1. Arc recognition smooths the arc, in this process any arc with a slope is curved and it does not provide the centerline

2. Text editor is not provided by this software

The result obtained by vectorizing the scanned map using WinTopo Professional is shown in figure 2.8.

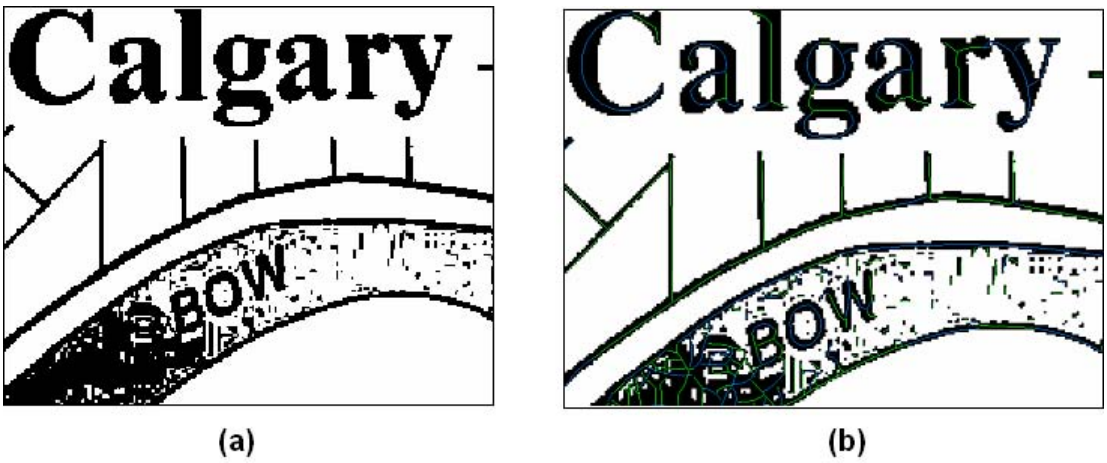

Figure 2.8: Wintopo Professional: a) Original Image, b) Result of Vectorization

Taking into account the above mentioned advantages and disadvantages we conclude that almost all the commercial software's require a perfect black and white scanned map to result in a good vector map. The quality of the output is directly proportional to the quality of the input. Editing raster data manually is much easier than editing vector data. Most of these software's use raster editing tools that use mathematical morphological operations like pruning, dilation, erosion, etc. Almost all the software have the despeckle filter in their raster editing toolbox. Despeckle

filter applies a soft, blurred effect to the image. It is especially effective on images with a high amount of contrast.

The next section gives a brief explanation of some of the available vectorization methods available in the field of digital image processing.

2.4 Automatic Raster to Vector Conversion Methods

Raster to vector conversion, consists of changing an image made up of cells (raster image) into one made up of points, lines and polylines (vector layer) [32]. Since it is natural to use the one pixel width skeleton or a set of medial points to simplify and represent a raster image, most vectorization methods are based on skeletonization methods or medial-axis approaches. Advanced vectorization includes line fitting and extending, which results in a complete accurate data set [63].

Image processing techniques were introduced more than 30 years ago and many basic vectorisation methods have been developed and implemented since then. These methods can be approximately divided into six classes [63]:

1. Hough Transform based methods
2. Thinning-based methods
3. Contour-based methods
4. Run-graph-based methods
5. Mesh pattern-based methods
6. Sparse-pixel-based methods

Methods	Time Complexity	Quality of Line Geometry	Line width preservation	Image Constraints
Hough Transform	quadratic	poor	no	Sparse, straight
Thinning	cubic	high	no	centerline
Contour	quadratic	poor	yes	straight
Run-graph	quadratic	poor	yes	straight
Mesh-Pattern	linear	poor	yes	Sparse, long
Sparse-Pixel	linear	good	yes	straight

Figure 2.9: Comparision of Vectorization methods (source: Wenyin, L. and Dori, D. (1999) [63])

Vectorisation is one of the most fundamental operation in recognition and interpretation of line drawings and document analysis. Choosing a vectorization method that best suits the needs of the system is very important. In general, good methods must preserve information like line width, line geometry and intersection junction as far as possible. Figure 2.9 summarizes the comparison of vectorization methods [63].

In the figure 2.9 we can see that hough transform based method is the fastest (as fast as Contour and Run graph based method) but the quality of the line that it produces is poor and it does not preserve the line width. Another limitation being that it results in sparse straight lines only. Thinning based method is faster than Mesh pattern and Sparse pixel based method but slower than Hough Transform nd Contour based method and run graph based method. The quality of the geometry of the line that it generates is the highest among all the other methods and it results in

the centerline which is very important for digitization. However, it does not preserve the line width.

Contour and Run graph based method are the fastest (as fast as Hough transform and Run graph based methods) and they also preserve the line width but, the quality of line geometry is poor and they result in straight lines only. Mesh pattern and Sparse pixel based method preserve the line width but they are the slowest and they do not preserve line geometry. Mesh pattern based method results in sparse long lines and Sparse pixel based method results in staright lines.

Since thinning based methods are comparatively faster and have high quality of line geometry and they result in the centerline of the object (lines) we choose this method for automatic digitization.

Thinning is an act of identifying a skeleton [42]. A map can be thinned or skeletonized using two methods, viz. Mathematical Morphology and Voronoi Diagrams. Both these methods are described in detail in the next chapter and some of the basic terminologies used in these methods are described in the next few sections.

The next section describes Mathematical Morphology and morphological operations.

2.5 Mathematical Morphology

Mathematical morphology is a method developed by Matheron and Serra [47] for quantitative analysis of spatial structures that aims at analyzing shape and forms of the object which originate from a set theory approach. Based on a formal mathematical framework, mathematical morphology provides an approach to the processing

of digital images that is based on geometrical shape. It uses set operations such as union, intersection and complementation [17], [47] and [42].

The basic idea behind digital morphology is that an image consists of a set of pixels that can be collected into groups that have a two dimensional structure or shape. For example, a line in an image is a collection of pixels that are connected and has a shape. By applying certain mathematical operations on the set of pixels, a specific aspect of the shape can be enhanced. Thus, for example, the number of pixels can be counted or a set of pixels can be recognized by its shape. The basic operations in mathematical morphology are erosion, in which pixels matching a given pattern are deleted from the image, and dilation, in which a small area around the pixel is set to a given pattern.

2.5.1 Structuring Element

The field of mathematical morphology provides a number of important image processing operations. All these morphological operators take two data input. One is the input image, which may be either binary or grayscale for most of the operators and the other is the structuring element. It is this element that determines the effect of the operator on the image.

The structuring element consists of a pattern specified as the coordinates of a number of discrete points relative to some origin. Normally, Cartesian coordinates are used and so a convenient way of representing the element is as a small image on a rectangular grid. Figure 2.10 shows a number of different structuring elements of various sizes. In each case the origin is marked by a cross.

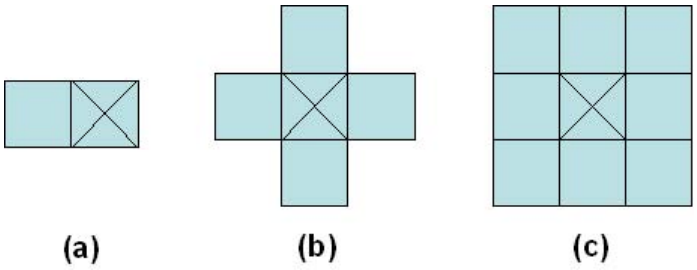

Figure 2.10: Structuring Elements: a) Right-neighbor Correlation, b) 5 Cross, c) 3 X 3

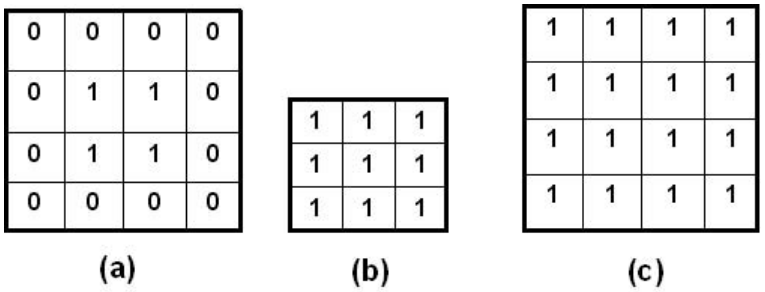

Figure 2.11: Binary Dilation: a) Original Image, b) Structuring Element, c) Dilation

2.5.2 Dilation

Dilation is one of the basic operators in the area of mathematical morphology. The basic effect of the operator on a binary image is to gradually enlarge the boundaries of regions of foreground pixels [42]. Thus areas of foreground pixels grow in size while holes within those regions become smaller. Binary dilation is shown in figure 2.11 and figure 2.13. In figure 2.11, (a) is the Original Image, (b) is the structuring element of size 3x3 and (c) is Dilation of the Original Image by the structuring element.

Dilation of the object A by the structuring element B in one dimension is given

by,

$$D(A,S) = A \oplus S = \{x : (\widehat{S})_x \cap A \neq \varnothing\} = \bigcup_{x \in S} A_x \tag{2.1}$$

where A and S are sets in Z. This definition is also known as **Minkowski Addition**. This equation simply means that the structuring element S is moved over A and the intersection of S reflected and translated with A is found. This equation is used to process binary sets of data.

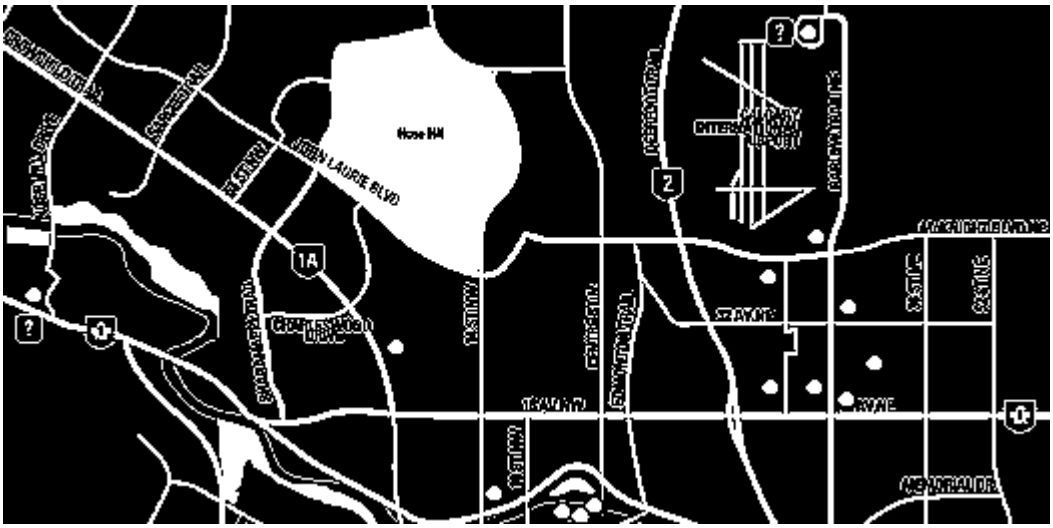

Figure 2.12: Scanned Black and White Map

2.5.3 Erosion

Erosion is one of the basic operators in the area of mathematical morphology. The basic effect of the operator on a binary image is to erode away the boundaries of regions of foreground pixels. Thus areas of foreground pixels shrink in size, and holes within those areas become larger [42]. Binary erosion is shown in figure 2.14 and figure 2.15. In figure 2.14, (a) is the Original Image, (b) is the structuring element of size 3x3 and (c) is Erosion of the Original Image by the structuring element.

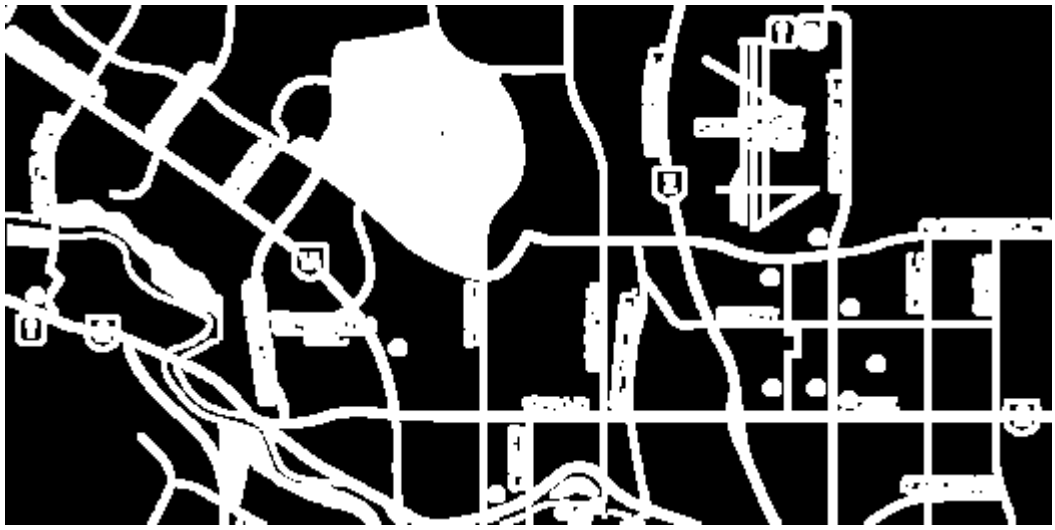

Figure 2.13: Binary Dilation

0	0	0	0	0
0	1	1	1	0
0	1	1	1	0
0	1	1	1	0
0	0	0	0	0

(a)

1	1	1
1	1	1
1	1	1

(b)

0	0	0	0	0
0	0	0	0	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0

(c)

Figure 2.14: Binary Erosion: a) Original Image, b) Structuring Element, c) Erosion

Erosion of the object A by the structuring element B in one dimension is given by,

$$E(A,S) = A \ominus S = \{x : (B)_x \subseteq A\} = \bigcap_{x \in S} A_x \tag{2.2}$$

where A and S are sets in Z. This definition is also known as **Minkowski Subtraction**. The equation simply says that erosion of A by S is the set of points x such that S translated by x is contained in A.

Figure 2.15: Binary Erosion

2.5.4 Opening

Erosion followed by dilation using the same structuring element is called opening [42]. It is less destructive than erosion. Opening an image A with the structuring element S can be mathematically represented as:

$$A \circ S = (A \ominus S) \oplus S \tag{2.3}$$

Opening makes a gap. All the elements that are shorter than the structuring element are removed, but the rest of the image is unchanged. This is a very useful property because if the filter is applied once then no more changes can be made to the result by applying repeated Openings [47]. This property is known as Idempotent.

$$(A \circ S) \circ S = A \circ S \tag{2.4}$$

An example of binary opening is shown in figure 2.16

Figure 2.16: Binary Opening

2.5.5 Closing

Dilation followed by erosion using the same structuring element is called closing [42]. It is less destructive than dilation. Closing an image A with the structuring element S can be mathematically represented as:

$$A \bullet S = (A \oplus S) \ominus S \tag{2.5}$$

Closing closes a gap. Closing also has the property of being idempotent [47].

$$(A \bullet S) \bullet S = A \bullet S \tag{2.6}$$

An example of binary closing is shown in figure 2.17.

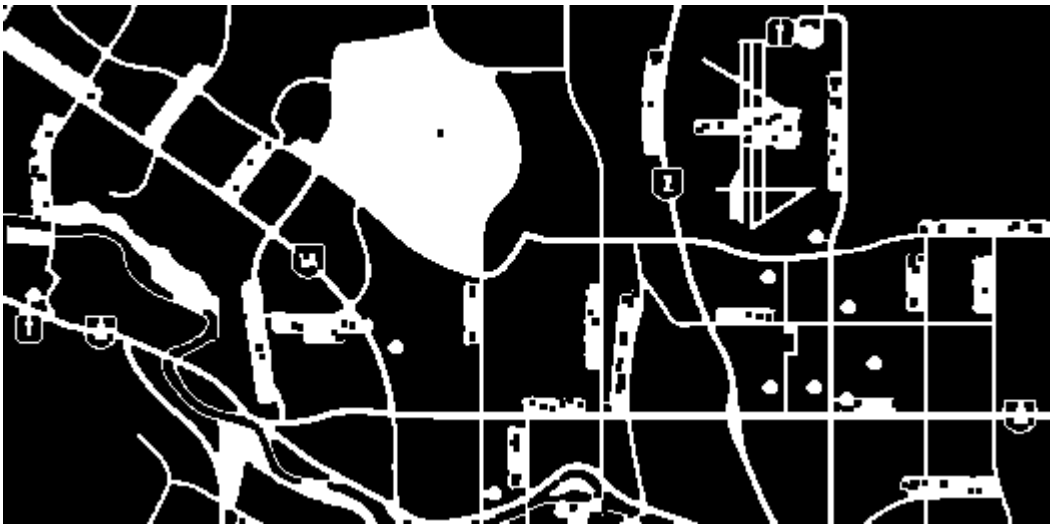

Figure 2.17: Binary Closing

2.5.6 Thinning

Thinning is used to remove selected foreground pixels from binary images, somewhat like erosion or opening. It is particularly used for skeletonization [42]. Skiena gives a very illustrative definition of the skeleton which is given by the prairie-fire analogy [49]. Consider the boundary of an object to be set on fire, the skeleton is the loci where the fire fronts meet and quench each other. Imagine a fire along all edges of the polygon, burning inward at a constant speed. The skeleton is marked by all points where two or more fires meet [49]. It is usually applied to binary images which produce another binary image as output.

2.5.7 Pruning

Skeletons produced by thinning often contain undesirable short spurs produced by small irregularities in the boundary of the original object [42]. These spurs can be removed by a process called pruning. This is shown in figure 2.18. In figure 2.18 (a) is the original image, (b) is the result of thinning and (c) is the result of pruning the original image.

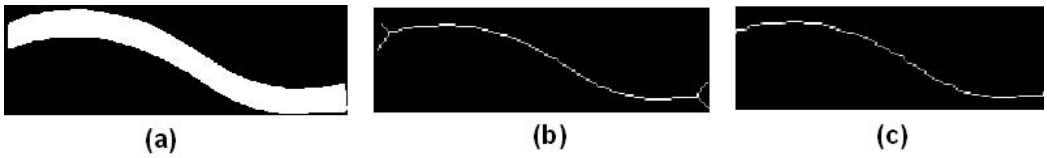

Figure 2.18: Pruning: a) Original Image, b) Thinning, c)Pruning

The next section describes how connectivity number can be calculated.

2.5.8 Connectivity Number

Thinning should not change the continuity of the original object by creating discontinuities or holes in the object (lines). Pruning is done in order to detect end points and remove the spurs pixel by pixel. A pixel is an end point if it is connected to just one pixel. If the end point has to be deleted then any line or an open curve will be completely deleted. Also, at times a single pixel connects two large sections of an object and obviously such a pixel should not be removed or deleted [42]. In order to prevent such deletion we measure the connectivity number and decide if the pixel should be deleted or not.

A connectivity number is a measure of how many objects (pixels) a particular pixel might be connected to. One such connectivity measure can be computed by

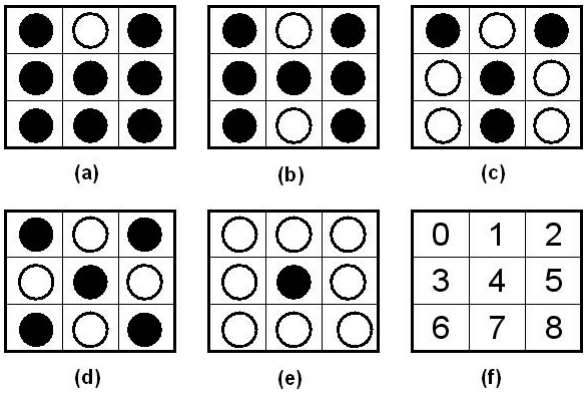

Figure 2.19: An Illustration of the connectivity number, a) The center pixel does not connect any region so it can be deleted. Connectivity number = 1, b) The center pixel cannot be deleted because the left and the right half would get disconnected. Connectivity number = 2, c) Connectivity number = 3, d) Connectivity number = 4 (maximum), e) Connectivity number = 0

visiting the neighbors in clockwise or anticlockwise order and counting the color changes. This number represents the number of pixels the center pixel is connected to. Simple points, like points for which the connectivity number is equal to one, and are not end points can be safely removed, because they do not belong to the skeleton. Figure 2.19 gives an illustrative example of calculating a connectivity number.

Till now we have seen how mathematical morphological operations like dilation, erosion, opening and closing can be applied on binary (black and white) images. In the next section, we will see how these morphological operations can be extended to grayscale images.

2.6 Grayscale Mathematical Morphology

The use of gray levels introduces an enormous complication, both conceptually and computationally. The pixels now has an integer value thus, the nice picture of the

image being a set disappears. Morphological concepts can be extended to grayscale images. The common approach is to use a grayscale extremum operation over a small neighborhood defined by the structuring element, i.e. grayscale morphology is simply a generalization of binary to images with multiple pixel, where the max and min operations are used in place of the OR and AND operations, respectively, of binary morphology [47] and [23].

All the binary morphological operations described in the previous section with the exception of thinning can be extended to grayscale images [23]. Grayscale thinning needs more research. A method introduced by John Weiss [60] and [61] for grayscale thinning of objects with one pixel width is shown in the next section.

2.6.1 Grayscale Thinning

Directly applying skeletonization on a grayscale map is a little complicated and there has not been much research done in this area. Binary morphological operations can be extended to grayscale using threshold superposition technique [61] and [60]. A grayscale image can be split into a series of binary images that correspond to binary thresholding at every intensity level. In threshold superposition, a grayscale image is seen as the sum of these binary images. Thus an 8-bit grayscale image is split into 255 binary images, by thresholding at intensity levels 1,2,3..., 255 [61] and [60]. The grayscale image can be recovered by summing up the binary images.

This is computationally intensive and takes roughly takes I times more time for a grayscale image with M intensity levels than for a binary image. Thus for an $M \times N$ grayscale image with M intensity levels, the binary operation must be applied $I \times M \times N$ times. For example, let us consider a grayscale image with three

intensity levels i.e. $I = 3$ of size 43×57 i.e. $M = 43$ and $N = 57$. Therefore the binary operation is applied 7353 times for this image.

The original image is shown in figure 2.20. The binary images and their respective skeletons are shown in figure 2.21 and the result of grayscale thinning using this technique is shown in figure 2.22. Here we can see that the result produced is not meaningful as we are interested in extracting the skeletons of specific gray values (selected features) and not all.

Figure 2.20: Original Image with Three Intensity Values

John Weiss [60] and [61] has also designed a template that can be used to detect the skeleton if the width of the skeleton is fixed and small(one pixel width). The template is shown in figure 2.23. Every template is convolved with a 3×3 neighborhood about every pixel in the image [60] and [61]. The template with the maximum response gives the magnitude and the direction of the skeleton. The disadvantage of this method is that as the thickness of the line increases the number of templates also increase because templates are required for every possible width and orientation of the line. This method becomes impractical if the width of the line exceeds a few pixels [60] and [61]. The result of applying these templates to figure 2.20 is shown

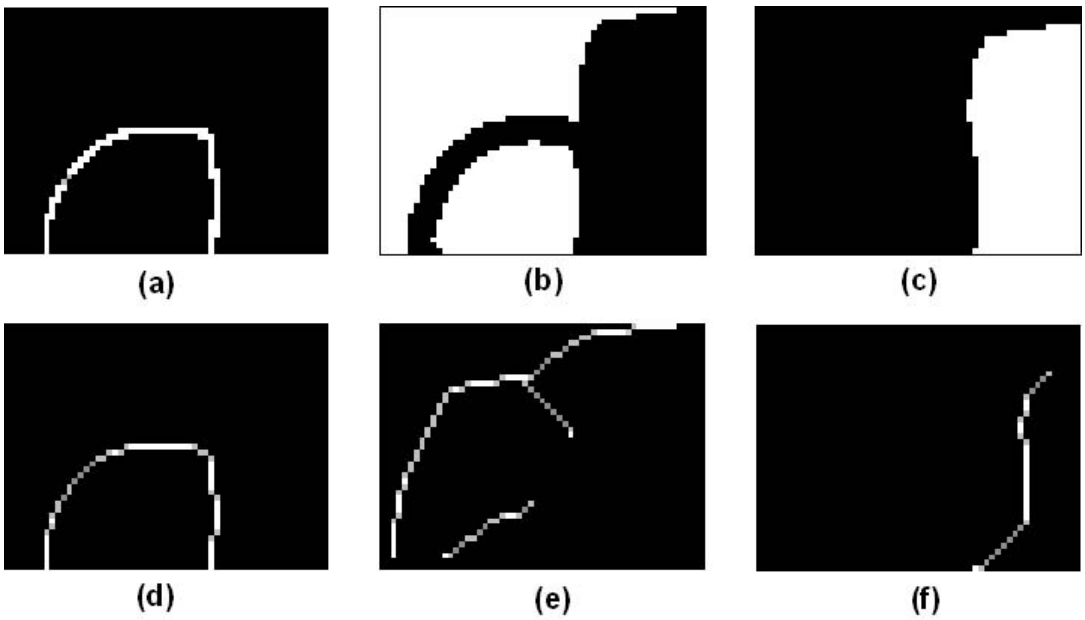

Figure 2.21: Grayscale thinning: (a), (b) and (c) are binary images with intensity values of 255, 143 and 204 respectively and (d), (e) and (f) are the corresponding skeletons

Figure 2.22: Addition of skeletons

in figure 2.24.

-1	-1	-1
2	2	2
-1	-1	-1

-1	-1	2
-1	2	-1
2	-1	-1

-1	2	-1
-1	2	-1
-1	2	-1

2	-1	-1
-1	2	-1
-1	-1	2

Figure 2.23: Template for Detecting a One-pixel-wide Ridge

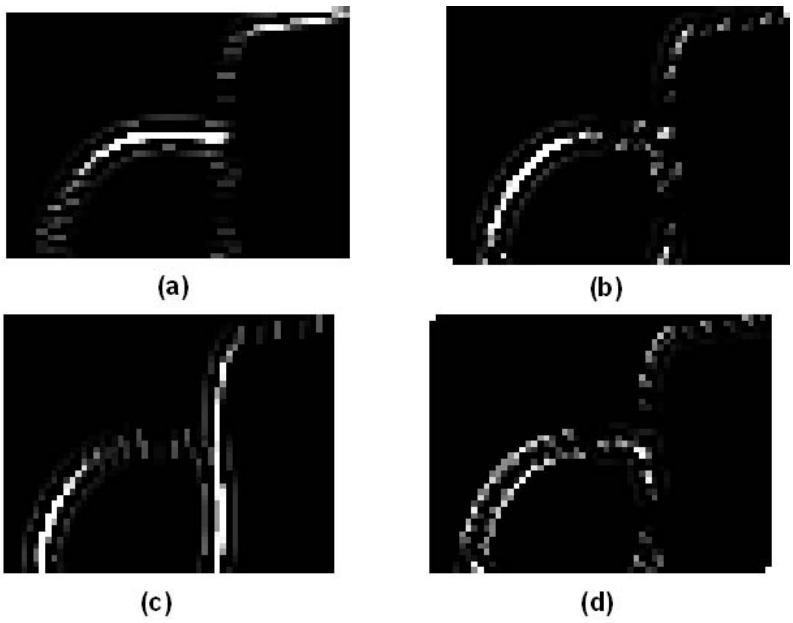

Figure 2.24: Grayscale thinning: (a), (b), (c) and (d) are skeletons generated by convolving the original image with template T1, T2, T3 and T4 respectively

The next section describes some of the terminologies used in thinning by Voronoi Diagrams.

2.7 The Voronoi Diagram

Let $\mathcal{O}$ be a set of sites in Euclidean space of dimension n. For each site o of $\mathcal{O}$, the Voronoi cell $V(o)$ of o is the set of points that are closer to o than to other sites of

$\mathcal{O}$. The Voronoi diagram $V(\mathcal{O})$ is the space partition induced by Voronoi cells [33]. An example of the Voronoi Diagram is shown in figure 2.25.

2.7.1 Delaunay Triangulation

The Delaunay triangulation is the dual of the Voronoi diagram. The Delaunay triangulation of $\mathcal{O}$ in the Euclidean space of n sites is the geometric dual of the Voronoi diagram of $\mathcal{O}$: two sites of $\mathcal{O}$ are linked by an edge in the Delaunay triangulation if and only if their cells are incident in the Voronoi diagram of $\mathcal{O}$ [33].

An example of Delaunay triangulation is shown in figure 2.25.

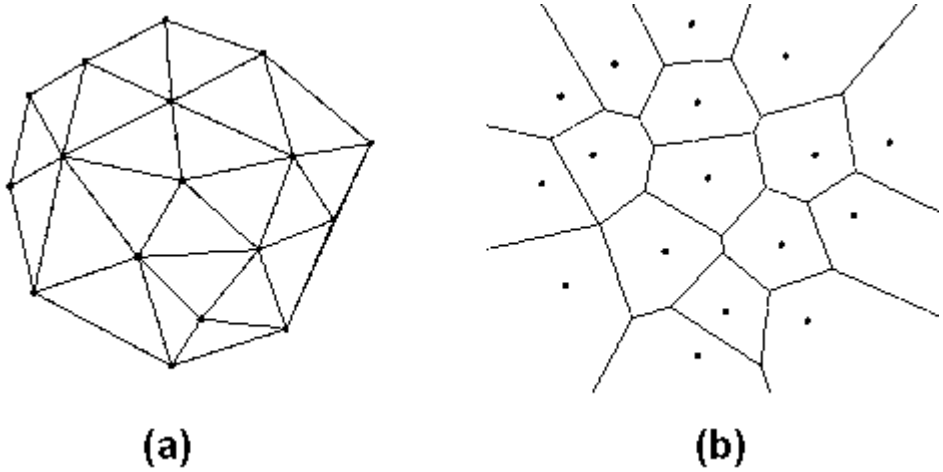

Figure 2.25: a) Delaunay Triangulation, b) Voronoi Diagram

Once the topological properties of the diagram are known, its geometrical properties like coordinates, lengths, angles, etc. can be computed in linear time depending on the number of sites. The computation of Voronoi Diagram can be greatly simplified by working with its dual, which is known as the Delaunay triangulation of the given sites [26]. This allows a cleaner separation between the topological and geometrical aspects of the problem.

Need for Data structure

Voronoi Diagram of a set of generating points can be simply drawn on a piece of paper. However, the drawn diagram or a set of numerical data equivalent to the drawn diagram does not convey enough information for many applications [39]. Assume that a computer program gives a list of line segments corresponding to the Voronoi edges as output. It is easy to draw the diagram using these edges but it is not easy to retrieve information such as a set of Voronoi polygon adjacent to one Voronoi polygon etc. Thus just drawing the Voronoi Diagram is not enough what we need is a data structure from which we can extract a variety of information necessary for applications.

In this research we have chosen to use the quad edge data structure that was developed by Guibas and Stolfi as a primitive topological structure for the representation of any subdivision on a two dimensional manifold [26].

2.7.2 Quad edge data structure

The quad edge data structure is the implementation of edge algebra, which is a mathematical data structure that defines the topology of any pair of dual subdivision on a two dimensional manifold [26]. A subdivision S of a manifold M is a partition of M into three finite collections of disjoint parts, the vertices, the edges, and the faces with the following properties:

1. Every vertex is a point of M.

2. Every edge is a line of M.

3. Every face is a disk of M.

4. The boundary of every face is a closed path of edges and vertices.

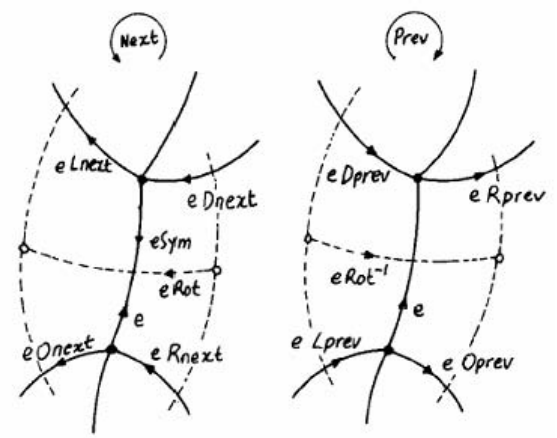

Figure 2.26: Edge function (Source: [26])

A directed edge of a subdivison P is an edge P with direction. Since direction and orientation can be chosen independently for every edge of a subdivision there are four directed and oriented edges. For any directed oriented edge we can define the following [26]:

- Vertex of origin : *e Org*
- Destination : *e Dest*
- Left face: *e Left*
- Right face : *e Right*
- Flipped edge e : *e Flip* (edge e with same direction, opposite orientation)
- Symmetric of e : *e Sym* (edge e with opposite direction, same orientation)

- The next edge with the same origin: *e Onext* (the edge immediately following e in counterclockwise direction in the ring of edged out of the origin of e)
- The next counterclockwise edge with the same left face: *e Lnext* (the first edge encountered after e when moving along the boundary of the face F)
- Rotate 90 degree in anticlockwise direction around crossing point: *eRot*

The next subsection describes the Quad-edge based Voronoi spatial data structure.

Quad-Edge based Voronoi data structure

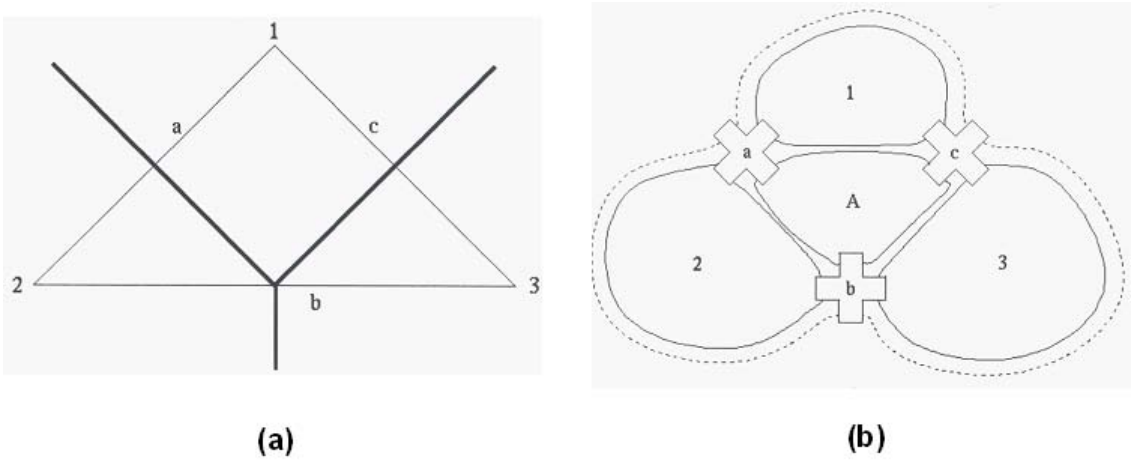

Figure 2.27: a) Voronoi diagram (thick line) of points 1, 2 and 3, b) Corresponding Quad edge data structure (source: [34])

As shown in the figure 2.28 each branch of the Quad-Edge (a,b,c) is a part of a loop around a Delaunay vertex/Voronoi face, or around a Delaunay triangle/Voronoi vertex (1,2,3). The quad edge data structure represents a graph (Voronoi diagram) and its geometric dual (Delaunay triangulation). Each quad edge has the following [26]:

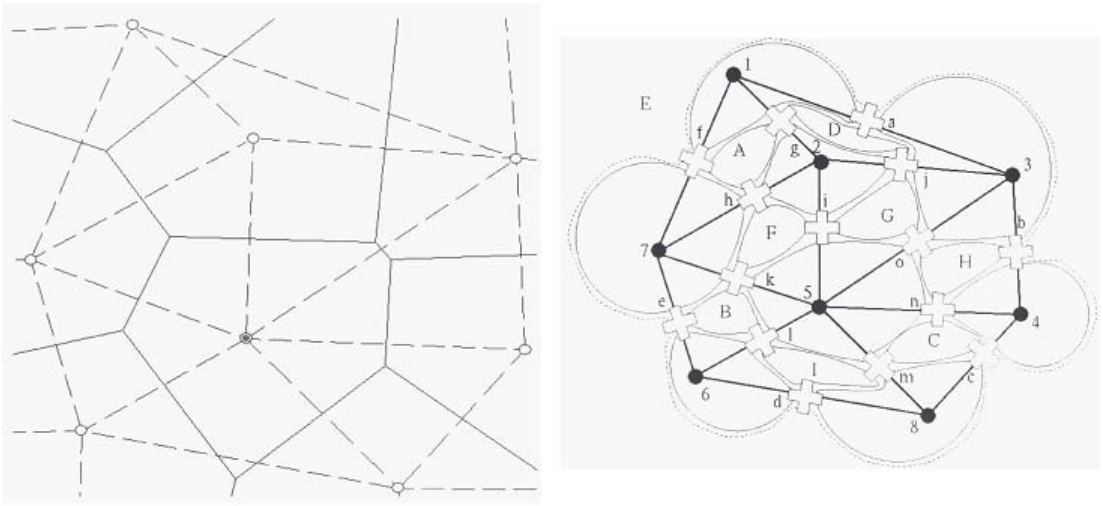

Figure 2.28: a) Voronoi diagram and Delaunay Triangulation, b) Corresponding Quad Edge Data structure (source: [34])

- Two pointers to the next Quad-Edge in counterclockwise order on the closed circuit around a vertex object
- Two pointers to the next Quad-Edge in counterclockwise order on the closed circuit around a face
- Two pointers to vertex objects
- Two pointers to faces

The edges can be partitioned into groups of eight. Each group consists of four oriented directed versions of an undirected edge of the subdivision of the manifold and plus four versions of its dual edge. The group containing a particular edge e is the orbit of e under the subalgebra generated by Rot and Flip. To build the data structure Guibas and Stolfi arbitrarily select a canonical representative in each

group. Then any edge e can be mathematically represented as $e_0 Rot^r Flip^f$, where $r \in 0, 1, 2, 3, f \in 0, 1$ and e_0 is the canonical representative of the group to which e belongs [26].

The group of edge containing e is represented by one edge record e, which is divided into 4 parts $e[0], e[1], e[2] and e[3]$. Part $e[r]$ corresponds to the edge $e_0 Rot^r$. an edge is represented by e, r and f which is called the edge reference. The edge reference also acts as a pointer to the quarter record $e[r]$.

Each quarter record of an edge record contains two fileds *Data* and *Next*. The Data field has geometrical and nontopological information and the Next field has information regarding edge reference and edge functions.

In the quad edge data structure a vertex is represented by a ring of edges and a standard way of representing it is by specifying one of its outgoing edges. A standard way of referring to a connected component of the edge structure is by giving one of its directed edges. This way one of the two dual subdivisions and a starting place and a starting direction is specified. Therefore, a edge e can be transformed to its dual by $eRot$.

The storage space required by the Quad edge data structure is equal to the number of edges multiplied by (8 record pointers + 12 bits). Each edge is connected to (immediately adjacent) Onext, Oprev, Dnext and Dprev and the 4 data fields corresponding to vertex and face links.

The advantage of using the quad edge data structure is that it allows uniform access to dual and mirror image subdivisions. It cuts in to half the number of primitives and derived operations since dual plus original is equal to a pair.

The next section describes the three methods of generating Voronoi Diagrams.

2.7.3 Methods for generating Voronoi diagram

Voronoi Diagrams can be constructed by the following three methods:

1. Incremental Method,
2. Plane Sweep Method and
3. Divide-and-Conquer Method.

We have chosen to use the incremental method because it is one of the most popular methods for building the Voronoi diagram, it is conceptually simple to implement and its average time complexity can be reduced to $O(n)$. The program coded by Dr. Francois Anton and Dr. Darka Mioc is used in this research.

Incremental Method

The incremental method was introduced by Green and Sibson [24]. This method starts with a simple Voronoi diagram for three sites and modifies the diagram by adding a new point one at a time. This process continues until the Voronoi diagram of N points are completed through the Voronoi diagram of N-1 points by adding the last point [39] and [35].

For example, let us consider that we have already constructed the Voronoi diagram of a set of points P_1, P_2, P_3, P_4 and P_5 as shown in figure 2.29. Now, we want to add a generator point P. First, we find the Voronoi polygon where P is inserted and then draw a perpendicular bisector between P and the generator point of that polygon [39]. In our case we draw a perpendicular bisector between P and $P1$ as shown in figure 2.30. The bisector crosses the boundary of the voronoi polygon $P1$ at two points; let the two points be W_1 and W_2 in such a way that P is to the

left of the directed line W_1W_2. Now, we grow the boundary of the Voronoi polygon P by drawing bisectors between the point P and the other points P_2, P_3, P_4 and P_5 in anticlockwise order [39]. The polygon formed ($W_1W_2W_3W_4W_5$) is the Voronoi polygon of P. This procedure is called the boundary growing procedure. The final result of inserting a new generator point is shown in figure 2.31.

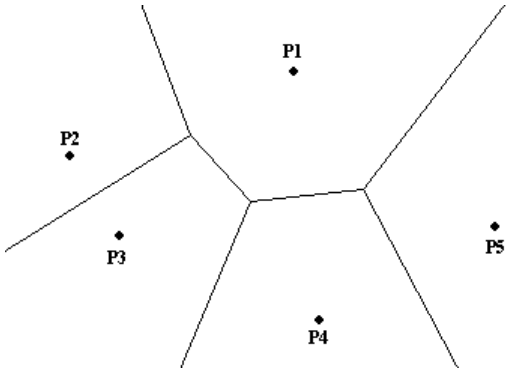

Figure 2.29: Voronoi Diagram of a set of generator points (P_1, P_2, P_3, P_4 and P_5)

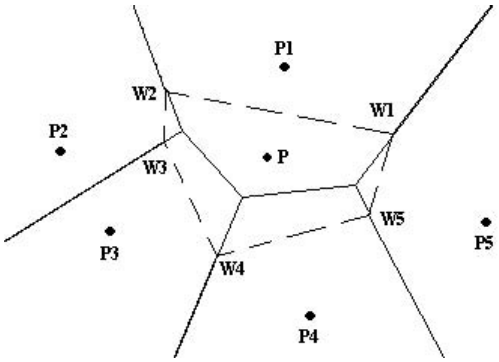

Figure 2.30: Addition of a new generator point P (Source: adopted from [39])

Thus in this section we have seen how Voronoi Diagrams can be constructed using the Incremental method. The process of extracting the skeleton from the Voronoi Diagram is shown in the next chapter. In skeletonization using Voronoi Diagrams we choose the data points as the difference in the level of gray. This is one of the main

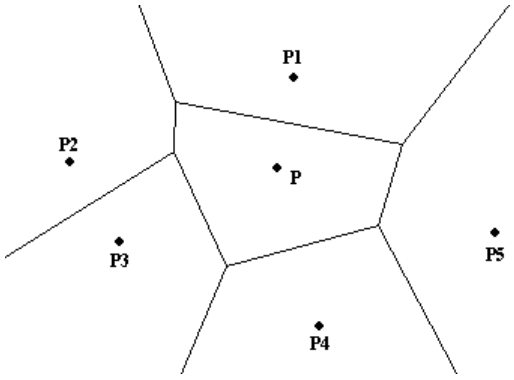

Figure 2.31: Final result of adding a new generator point

advantage of this method. Due to this reason we can skeletonize black and white as well as grayscale images. The methodology is explained in detail in the next chapter.

A grayscale image can be skeletonized in two ways. One way would be to convert it to a binary image and then skeletonize it and the other way would be to directly apply the algorithm for grayscale skeletonization. A grayscale image can be converted to a binary image by thresholding. The next section describes how thresholding can be done.

2.8 Thresholding

Thresholding is used to separate the regions of the image corresponding to an object from its background. Thresholding is a very convenient way of performing segmentation. Here, we use the difference in intensities or colors in the foreground and background regions of an image to generate a binary image [23] and [42]. Thus thresholding can be defined as an image processing function that is used to segment an image by setting all pixels whose intensity values are above a threshold to a foreground value and the remaining pixels to a background value [23]. It can also be

used to see what areas of an image consist of pixels whose values lie within a specified range, or band of intensities (or colors).

The input to a thresholding operation is typically a grayscale or color image. In the simplest implementation, the output is a binary image representing the segmentation. Black pixels correspond to background and white pixels correspond to foreground (or vice versa). In simple implementations, the segmentation is determined by a single parameter known as the intensity threshold. In a single pass, each pixel in the image is compared with this threshold. If the pixel's intensity is higher than the threshold, the pixel is set to white in the output. If it is less than the threshold, it is set to black [23], [42] and [17]. This method is called global thresholding.

In more sophisticated implementations, multiple thresholds can be specified, allowing a band of intensity values to be set to white while everything else is set to black. For color or multi-spectral images, it may be possible to set different thresholds for each color channel, and so select just those pixels within a specified cuboid in the RGB space. Another common variant is to set to black all those pixels corresponding to background, but leave foreground pixels at their original color/intensity (as opposed to forcing them to white), so that the information is not lost [23], [42] and [17]. This method is called local thresholding.

Not all images can be segmented into foreground and background using simple thresholding. We can check if the image can be neatly segmented or not, by looking at the intensity histogram of the image. If it is possible to separate out the foreground of an image on the basis of pixel intensity, then the intensity of pixels within foreground objects must be distinctly different from the intensity of pixels within

the background. In this case, we expect to see a distinct peak in the histogram corresponding to foreground objects such that thresholds can be chosen to isolate this peak accordingly. If such a peak does not exist, then it is unlikely that simple thresholding will produce a good segmentation [23], [42] and [17]. In this case, local thresholding may be a better answer.

An image composed of light objects on a dark background mostly corresponds to a bi-model histogram This image can be successfully segmented using a single threshold T, that separates these modes. Any point that is greater than or equal to T can either be the object or the background. This approach is called global thresholding.

However, for some images the histograms are slightly more complicated. It is almost certainly not possible to successfully segment this image using a single global threshold. The histogram can have three or more peaks, so we have to allow the threshold to vary. This approach is called local thresholding.

2.8.1 Global Thresholding

One way of choosing a threshold is by visually inspecting the image histogram as shown earlier. Another way of choosing a threshold is by trial and error, i.e. by picking different thresholds until one is found that produces good result as judged by the observer. There are different methods for automatically choosing a threshold [40] and [23]. One such method that is histogram based is Otsu's method. The methods is described as follows:

1. The normalized histogram is treated as a discrete probability density function

$$P_r(r_q) = \frac{n_q}{n} \qquad q = 0, 1, 2, ..., L-1 \tag{2.7}$$

where n is the total number of pixels in the image, n_q is the number of pixels that have intensity level r_q and L is the total number of possible intensity levels in the image.

2. Now, suppose that a threshold K is chosen such that C_0 is the set of pixels with levels $[0, 1, ...K-1]$ and C_1 is the set of pixels with levels $[K, K+1, ..., L-1]$ Otsu's method chooses the threshold value K that maximizes the between-class variance $(\sigma_B)^2$, which is defined as [23] and [40]:

$$(\sigma_B)^2 = \omega_0(\mu_0 - \mu_T)^2 + \omega_1(\mu_1 - \mu_T)^2 \tag{2.8}$$

$$\tag{2.9}$$

where

$$\omega_0 = \sum_{q=0}^{k-1} P_q(r_q) \tag{2.10}$$

$$\omega_1 = \sum_{q=0}^{L-1} P_q(r_q) \tag{2.11}$$

$$\mu_0 = \sum_{q=0}^{K-1} \frac{qP_q(r_q)}{\omega_0} \tag{2.12}$$

$$\mu_1 = \sum_{q=k}^{L-1} \frac{qP_q(r_q)}{\omega_1} \tag{2.13}$$

$$\mu_T = \sum_{q=0}^{L-1} qP_q(r_q) \tag{2.14}$$

$$\tag{2.15}$$

3. In this methods we first compute the histogram and then find the threshold value that maximizes $(\sigma_B)^2$.

4. The threshold is returned as a normalized value that lies between 0.0 and 1.0

5. Using this value we threshold the image and convert the grayscale image to black and white.

The result of thresholding a grayscale map (figure 2.32) using Otsu's algorith is shown in figure 2.33.

Figure 2.32: Scanned Grayscale Map

The result of applying this method on a satellite imagery is shown in figure 2.35. Figure 2.34 is the original image and figure 2.35 is the thresholded image.

2.8.2 Local Thresholding

Local thresholding is also called as adaptive thresholding or dynamic thresholding. The conventional thresholding operator uses a global threshold for all pixels whereas

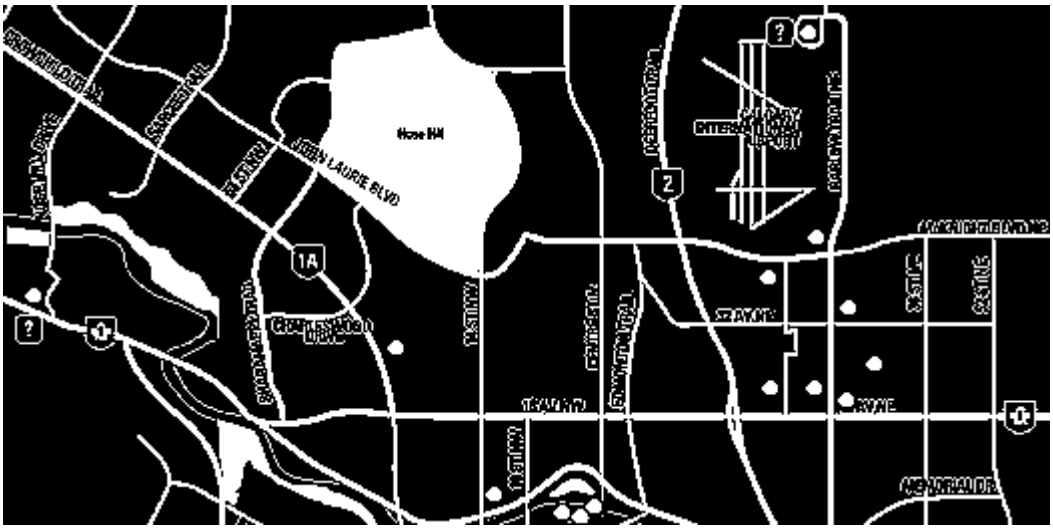

Figure 2.33: Thresholded Image

Figure 2.34: Satellite Imagery

Figure 2.35: Global Thresholding using Otsu's algorithm

adaptive thresholding changes the threshold dynamically over the image. This more sophisticated version of thresholding can accommodate changing lighting conditions (uneven illumination) in the image, for e.g. those occurring as a result of a strong illumination gradient or shadows [23] and [17].

A common practice in such case is to preprocess the image to compensate for the illumination problem and apply a global threshold to the preprocessed image. This could be done by applying a morphological top-hat operator and then using Otsu's method on the result [23].

An alternative approach for finding the local threshold is to statistically examine the intensity values of the local neighborhood of each pixel. The statistic which is most appropriate depends largely on the input image. Simple and fast functions include the mean of the local intensity distribution, where T can be the mean, the median or the mean of the minimum and the maximum values [17].

The size of the neighborhood has to be large enough to cover sufficient foreground and background pixels, otherwise poor threshold is chosen. On the other hand, choosing regions which are too large can violate the assumption of approximately

uniform illumination [17].

2.9 Summary

In this chapter we have seen a review and evaluation of the existing spatial data applications for raster/vector conversion in GIS. Since it is important to review and evaluate the existing spatial data applications for raster/vector conversion in GIS, a brief idea of the vectorization software available today in the commercial world and their comparison with respect to the results it produces are given.

Since thinning based methods are comparatively faster and have high quality of line geometry and they result in the centerline of the object (lines) we chose this method for automatic digitization. Then, a description of previous research in thinning using Mathematical Morphology and Voronoi Diagram is given. We also chose the incremental method for generating Voronoi Diagram because it is one of the most popular methods for building the Voronoi diagram, it is conceptually simple to implement and its average time complexity can be reduced to $O(n)$. Finally different techniques for thresholding was shown in the last section.

The next chapter shows the methodology adopted to vectorize (automatically digitize) scanned maps using Voronoi Diagrams and Mathematical Morphology. This chapter shows how the centerline can be extracted from scanned maps using skeletonization by Mathematical Morphology and Voronoi diagram.

Chapter 3

Skeletonization

3.1 Introduction

Research on automatic vectorization has been going on for many years resulting in a large amount of publications. Recent publications on vectorization of line drawings have introduced the main steps of the automatic procedure, which are [45]:

1. Digitization of the original paper document using a scanner
2. Filtering
3. Thresholding
4. Thinning and Pruning the binary image
5. Raster to vector conversion

For the first four steps, different algorithms can be applied and good results can be obtained, but step 5 gives different results depending on the method that is applied [45]. Vectorization is basically the center line extraction of any feature. Centerlines can be extracted using Skeletonization. The two different techniques of extracting skeletons are, Skeletonization by Mathematical Morphology using Thinning and Voronoi Diagrams.

An image should be skeletonized by taking the following three criteria's into account:

1. Connectivity should be preserved i.e. if the object is connected in the original image then the resulting skeleton should also be connected. However, if the background is not connected then the background should not be connected after thinning.

2. Excessive erosion should be prevented.

3. Pruning should be done after the extraction of the skeletons. The skeleton of the object should not change due to noise or small convexities which do not belong to the object.

We can see that criteria 2 and 3 contradict each other. Usually certain constraints are added to make a compromise between the two criteria.

This chapter is divided into three sections. The first section describes the algorithm used for Skeletonization by Mathematical Morphology using thinning and the second section describes the algorithm used for Skeletonization by Voronoi Diagrams. The third section shows some of the applications of this research and gives a brief description of one application which is linear feature extraction from satellite imageries.

3.2 Skeletonization by Mathematical Morphology

3.2.1 Introduction

Thinning is one of the Mathematical Morphological operations that is particularly used for skeletonization [42]. It is used to remove selected foreground pixels from binary images, somewhat like erosion or opening. The methodology for skeletonization

of scanned maps using Mathematical Morphology is described below.

The paper map is scanned to produce a raster map, the raster map is then filtered to remove noise. In order to avoid the dithering effect the paper map should be scanned with the scanning software's dithering function switched off. There are a variety of filters available to remove noise in the field of Digital Image Processing. Median filter is used in this research since it takes one pixel at a time and looks at its neighbors to decide if or not it represents its surroundings and replaces the center pixel value with the median of its neighbors (including the center pixel) [17]. The median is calculated by first sorting all the pixel values from the surrounding neighborhood into numerical order and then replacing the pixel being considered with the middle pixel value. If the neighborhood under consideration contains an even number of pixels, the average of the two middle pixel values is used [17]. Then thresholding is done using Otsu's algorithm to highlight only the feature of interest in the map. Now, a clean binary image is ready for skeletonization using thinning.

3.2.2 Methodology

There are a variety of parallel thinning algorithms available in the field of digital image processing. All most all the algorithms are similar. We have chosen to use Zhang-Suen's algorithm for thinning because it has been used as a basis for comparison of thinning methods for many years and it is comparitively fast and simple to implement [42].

Skeletonization by thinning is done by using the following three steps:

1. Preprocessing the image with Stentifords acute angle emphasis preprocessing step

2. Applying Zhang-Suen's algorithm for thinning

3. Post processing using Holt's staircase removal algorithm.

The next section describes the algorithms used for preprocessing, thinning and post processing.

The Algorithm

Preprocessing

Thinning results in necking, tailing and spurious projects in the skeleton. They are called classical thinning artifacts (figure 3.1). A narrow point at the intersection of two lines stretched into a small line segment is called necking. Tails are created because of excess thinning, where two lines meet at an acute angle and finally the most common artifact is the creation of extra line segments joining a real skeleton which is called a spurious projection, hairs or line fuzz.

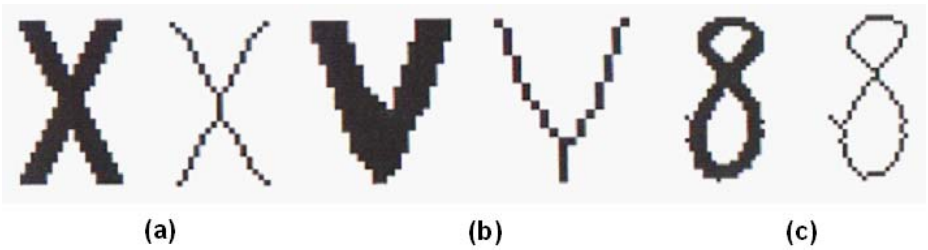

Figure 3.1: Classic thinning artifacts, a) Necking, b) Tailing, c) Spurious projection (source: [42])

Stentiford has suggested a preprocessing step to minimize these thinning artifacts [42]. Since line fuzz is usually caused by some small irregularities in the object outline, a smoothing step before thinning would help remove them. In this process a kernel is passed over the image and the center pixel having two or fewer black pixels

is deleted i.e. having a connectivity number of less than two. In this research since we pass a median filter to remove noise line fuzz has already been removed.

Stentiford suggests a method called acute angle emphasis to deal with necking. In this procedure the pixel near the joint between two lines are set to white if they form an acute angle. This is done using the template shown in the figure 3.2. A match to any template marks the center pixel for deletion and begins another iteration of a less severe acute angle emphasis using only the first three templates of each type (type D and type U). If any pixel is deleted then one last pass using only the first template of each type is performed [42].

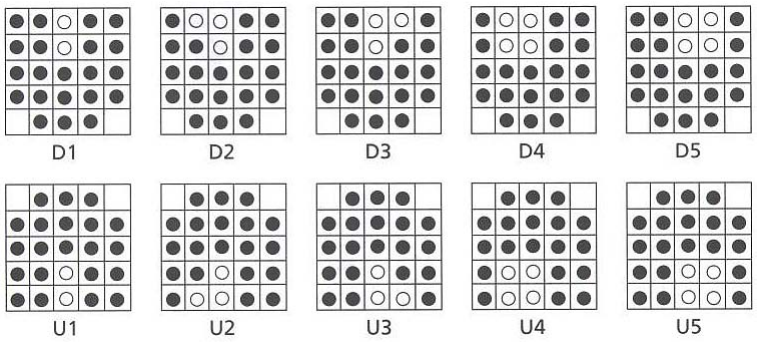

Figure 3.2: Templates used for acute angle emphasis preprocessing step (source: [42])

Thinning

Zhang-Suen's algorithm for thinning is a parallel method, meaning that the new value for any pixel can be computed only by using the values obtained from the previous iteration. The algorithm is divided into two sub iterations. In one sub iteration, a pixel I(i, j) is deleted (or marked for deletion) if the following four conditions are all true:

1. Its connectivity number is one.

2. It has at least two black neighbors and not more than six.

3. At least one of $I_{(i,j+1)}, I_{(i-1,j)}$ and $I_{(i,j-1)}$ are background (white).

4. At least one of $I_{(i-1,j)}, I_{(i+1,j)}$ and $I_{(i,j-1)}$ are background.

At the end of this sub iteration the marked pixels are deleted. In the next sub iteration a pixel I(i, j) is deleted (or marked for deletion) if the following four conditions are all true:

1. Its connectivity number is one.

2. It has at least two black neighbors and not more than six.

3. At least one of $I_{(i-1,j)}, I_{(i,j+1)}$ and $I_{(i+1,j)}$ are background.

4. At least one of $I_{(i,j+1)}, I_{(i+1,j)}$ and $I_{(i,j-1)}$ are background.

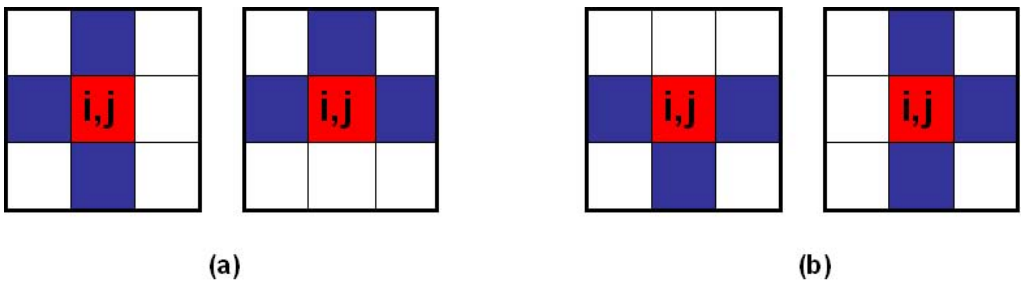

Figure 3.3: a) Pixel marked for deletion in 1st iteration, b) Pixel marked for deletion in 2nd iteration

The marked pixels are deleted. If at the end of either sub iteration there are no pixels to be deleted, then the skeleton is complete and the program stops [42].

Post Processing

In 1987, Holt [42] suggested an improvement to Zhang-Suen's algorithm that is faster and does not involve sub iterations. The first sub iteration can be written as a logical expression as,

$$v(C) \wedge (\sim edge(C) \vee (v(E) \wedge v(S) \wedge (v(N) \vee v(W)))) \tag{3.1}$$

Where the center pixel is not marked for deletion in the first sub iteration and the function v gives a value of the pixel, 1 is true for an object and 0 is false for the background. The edge function is true if C is on the edge of the object, i.e. having the neighbors between 2 and 6 and connectivity number = 1. E, S, N and W correspond to pixels in a particular direction from the center pixel C; E is east i.e. I (i, j+1), S is south i.e. I (i+1, j) and so on [42].

The second sub iteration can be written as a logical expression as,

$$v(C) \wedge (\sim edge(C) \vee (v(W) \wedge v(N) \wedge (v(S) \vee v(E)))) \tag{3.2}$$

Holt et. al. combined the above 2 expressions with a connectedness preserving condition for parallel execution into the following single expression [42]:

$$\begin{aligned} & v(C) \wedge (\sim edge(C) \vee \\ & (edge(E) \wedge v(N) \wedge v(S)) \vee \\ & (edge(S) \wedge v(W) \wedge v(E)) \vee \\ & (edge(E) \wedge edge(SE) \wedge edge(S)) \end{aligned} \tag{3.3}$$

At times even if thinning is complete, there are some pixels that could be deleted. The idea behind this is that the pixels that form a staircase are removed and we

know that half of the pixels that form a staircase can be removed without affecting the shape of the object. So, the center pixel of one of the windows shown in figure 3.4 can be deleted:

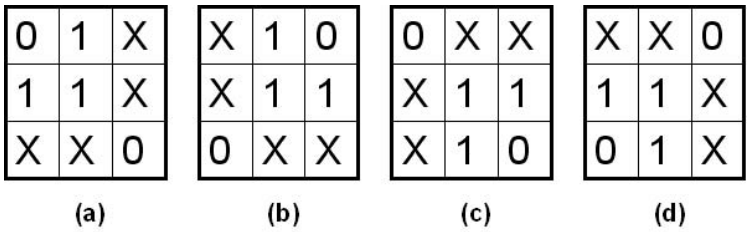

Figure 3.4: Holts Staircase Problem. The center pixel in one of the above can be deleted

In order to avoid creating a new hole, we add a simple condition that one of the values of X = 0 in figure 3.4. For kernels having a northward bias (figure 3.4: a, b) the expression to save a pixel from being deleted in the staircase removal iteration is:

$$\begin{aligned} & v(C) \wedge \sim (v(N) \wedge \\ & ((v(E) \wedge \sim v(NE) \wedge \sim v(SW) \wedge (\sim v(W) \vee \sim v(S)) \vee \\ & (v(W) \wedge \sim v(NW) \wedge \sim v(SE) \wedge (\sim v(E) \vee \sim v(S)))))) \end{aligned} \tag{3.4}$$

The pass having the southward bias is the same, but with north and south exchanged. However, this method does not remove tails and line fuzz [42].

Till now we have seen the methodology used for skeletonization using thinning by Mathematical Morphology. The next subsection shows the results obtained by appling this method on a binary scanned map and/or image.

3.2.3 Skeletonization of Grayscale Maps by Mathematical Morphology

The result of applying this algorithm on a scanned black and white map is shown in figure 3.5 and figure 3.7. In figure 3.5, (a) is a simple black and white image and (b) shows the result obtained by skeletonization using thinning. In the second example, the original image is shown in figure 3.6 and the skeleton obtained by the combining three algorithms is shown in figure 3.7.

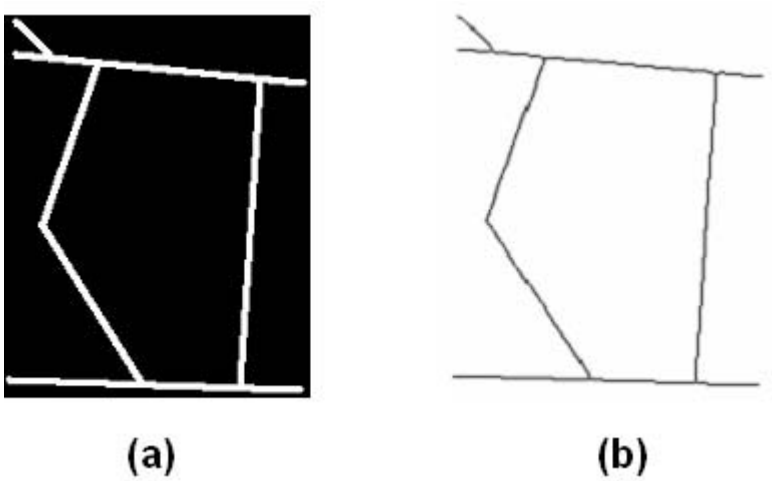

Figure 3.5: Experimental Results, a) Original Image, b) Skeleton

Figure 3.6: Scanned Map (Source: Library, University of Calgary)

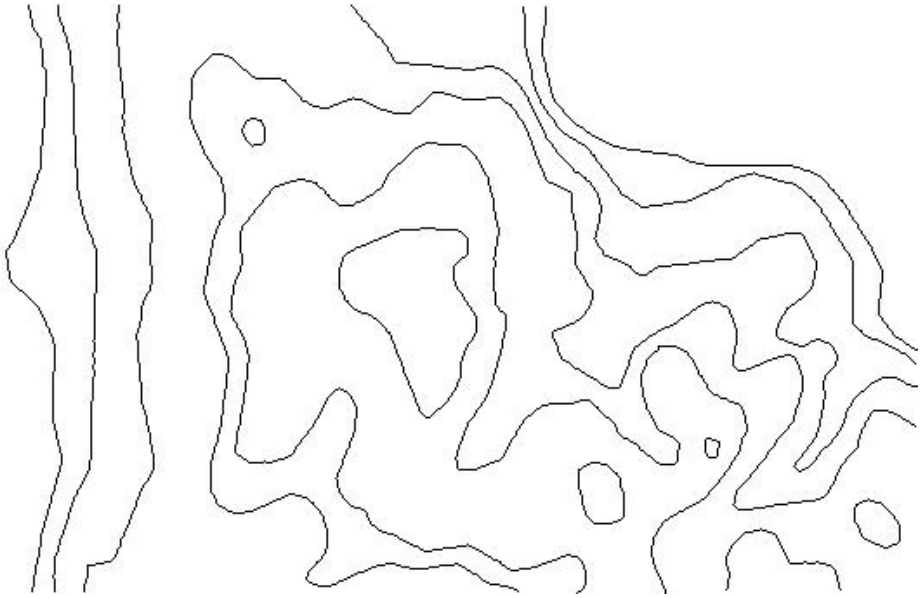

Figure 3.7: Skeletonization by Mathematical Morphology

Till now we have seen how a black and white scanned map is vectorized. In the next section we will see how a grayscale map is skeletonised.

3.2.4 Skeletonization of Grayscale Maps by Mathematical Morphology

Skeletonization of Grayscale Maps using Mathematical Morphology can be done in the following two ways:

1. Converting the grayscale map into binary map (black and white map) by thresholding and then applying the above mentioned method

2. Directly applying morphological operations on the grayscale image

The result of converting the grayscale map into binary map (black and white map) by thresholding is shown in figure 3.8. Applying thinning to the black and white image is shown in figure 3.9.

Figure 3.8: Grayscale map

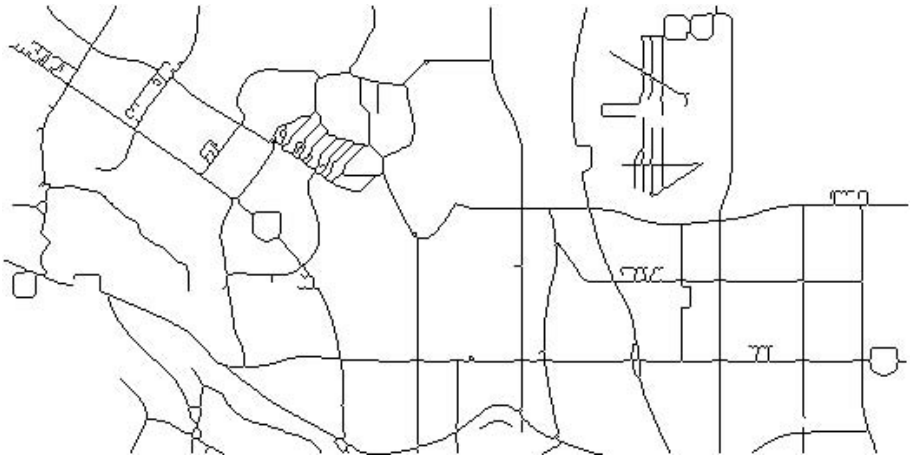

Figure 3.9: Skeletonization by Mathematical Morphology (Grayscale scanned map)

An approach for grayscale thinning has been proposed by John Weiss [60]. As shown in the previous chapter he has suggested four templates for detecting a one pixel width ridge. Each of the four template is convolved with the image. The result of convolving figure 3.8 with template T1 and T2 produce good result and the results are shown in figure 3.10 and 3.11.

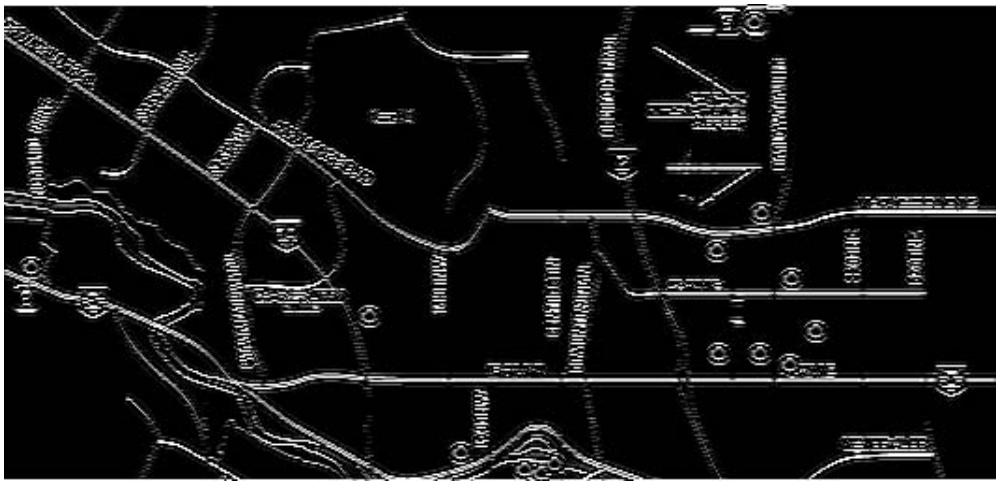

Figure 3.10: Grayscale thinning (convolved with template 1)

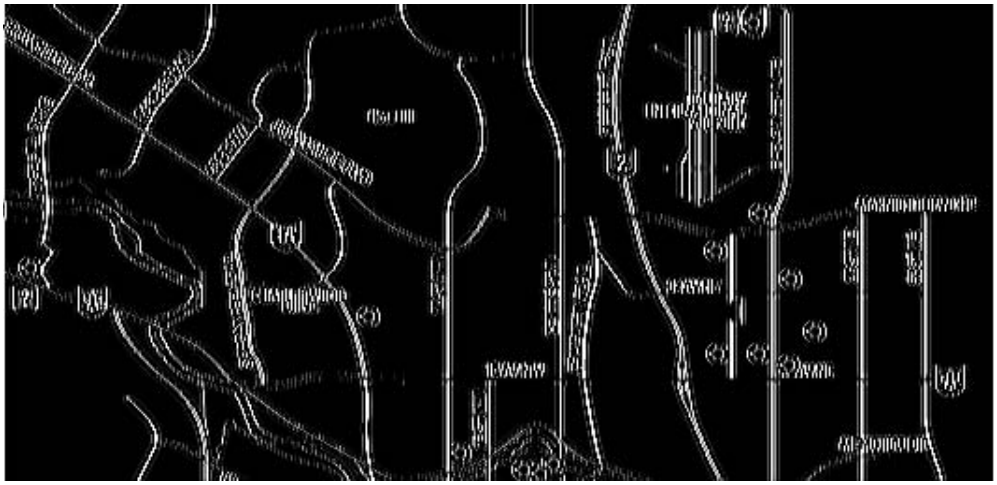

Figure 3.11: Grayscale thinning (convolved with template 3)

The next section describes the methodology of Skeletonization by Voronoi Diagrams.

3.3 Skeletonization by Voronoi Diagrams

3.3.1 Introduction

A good illustrative definition of a skeleton is given by the prairie-fire analogy which sates that if the boundary of an object is set on fire then the skeleton is the loci where the fire fronts meet and quench each other. Imagine fire along all edges of a stick (narrow rectangular polygon), burning inward at a constant speed. The skeleton [49] is marked by all points where two or more fires meet. This operation is also called as medial-axis transformation. Another approach to compute skeletons is based on the Voronoi diagram.

3.3.2 Methodology

As shown in section 3.2.1 the paper map is scanned and median filter is used to remove noise. Then Otsu's algorithm for thresholding is used to create a binary map. Then edge detection operation is applied on this map and every pixel on the edge is used as a data point to generate the Voronoi Diagrams and Delaunay Triangulation. We have chosen to generate Voronoi Diagram by incremental method because it is an efficient, powerful and popular method.

The next section describes the edge detection algorithm used for this method.

Edge Detection

Edge detection can be defined as an image-processing technique in which edge pixels are identified by examining their neighboring pixels [32].

We have chosen to use Laplacian for edge detection in this research. This is done in order to get sample points around the high frequency changes in the image. Moreover, Laplacian uses the second derivative where the edges occur at the peaks. In order to get the edge points we compute the laplacian and find the zero crossings.

The Laplacian $\left(\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}\right)$ is proportional to the variation of the derivative of the interpolant at the point with respect to an annulus centered at the point. Two different computations are used for the Laplacian, the standard Laplacian which uses a neighborhood of 4 pixels and alternative Laplacian which uses a neighborhood of 8 pixels. In alternative Laplacian, a $\frac{1}{\sqrt{2}}$ factor is used to compensate for the wider diagonal pixel separation. These two Laplacian based sampling techniques are the other two sampling techniques that are used to get the sample points. Alternative Laplacian gives better results than standard Laplacian.

Data points are selected where the variation of the intensity is the highest, i.e. at the edges. Sampling consists of selecting all the pixels whose derivative operator value is bigger than some threshold. After the image is irregularly sampled, the Voronoi Diagrams and its dual graph, the Delaunay triangulation of the set of sample points is computed using an incremental algorithm based on the Quad-Edge data structure.

The Algorithm

The algorithm that was developed by Amenta *et al.* and modified by Gold *et al.* and further modified by Mioc *et al.* for the skeletonisation of the raster image is

presented. The edges of the feature are taken as the boundaries of the zones where the level of gray changes continuously. The edges are first detected as a subset of the sample points that are used to generate the Voronoi Diagrams and the Delaunay triangulation. Then based on the edges of the Voronoi Diagrams of the sampled pixels, the edges are detected based on the following three criterion ,

1. The first criterion is that the difference of the levels of gray of the extremities of the Voronoi edge should be smaller than the difference in the levels of gray of the extremities of the dual Delaunay edge.

2. The second criterion is that the dual Delaunay edge and the Voronoi edge (considered as line segments) intersect.

3. The third criterion is that the levels of gray of the extremities of the Voronoi edge and of the dual Delaunay edge are not all the same.

The resulting subset of the Voronoi Diagrams is the set of all the edges of the picture, which is called as the border set. The Voronoi edges adjacent to a Voronoi edge of the border set that do not belong to the border set is flagged (marked). Then from this border set, the skeleton using a traversal of the Voronoi zones belonging to the interior of the border set is drawn (plotted). For each Voronoi edge e of the border set, the Voronoi edges of the Voronoi zone that belongs to the interior of the border set starting from the Voronoi edge following e is traversed.

Every traversed Voronoi edge is marked as visited. For each one of that Voronoi edges f that belongs to the border set and is not adjacent to e, the bisector of e and f is drawn in the skeleton. For each one of those Voronoi edges f that is not in the

border set and such that one of its neighbors is flagged, the Voronoi edge is drawn in the skeleton.

The next subsection shows how this method works on black and white scanned maps.

3.3.3 Skeletonization of black and white Maps by Voronoi Diagram

Applying the methodology described above the following results are obtained. Figure 5.1 is the original image, figure 3.13 is the corresponding Voronoi Diagrams, figure 3.14 is the corresponding Delaunay Triangulation (dual of Voronoi Diagram) and figure 3.15 is the Skeleton generated using Voronoi Diagram.

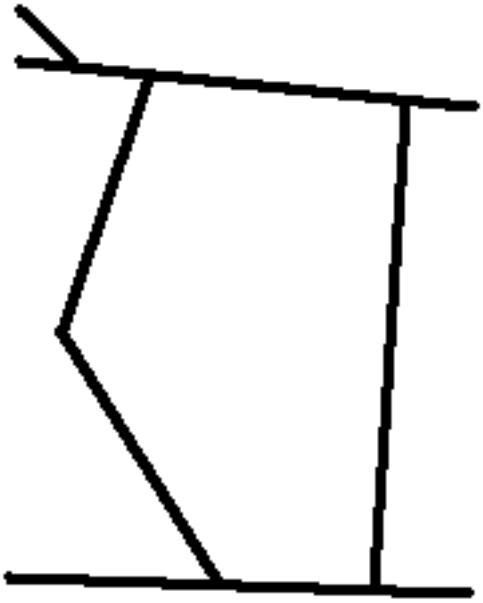

Figure 3.12: Original Image

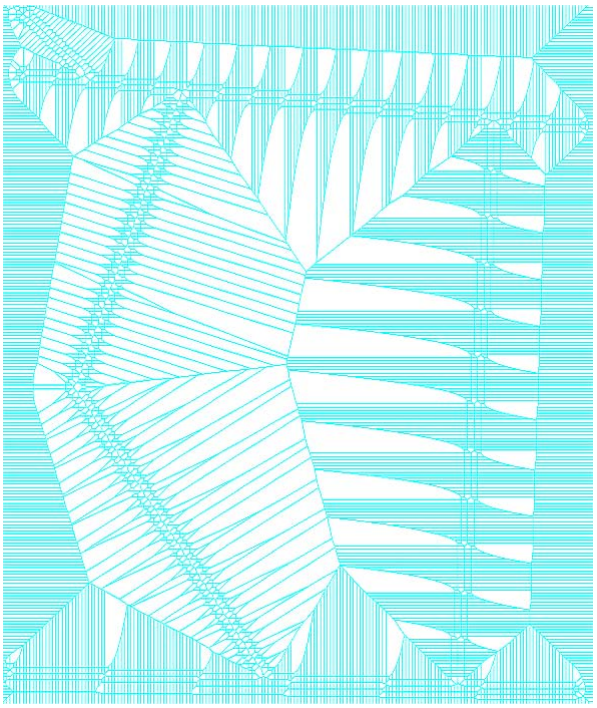

Figure 3.13: Voronoi Diagrams

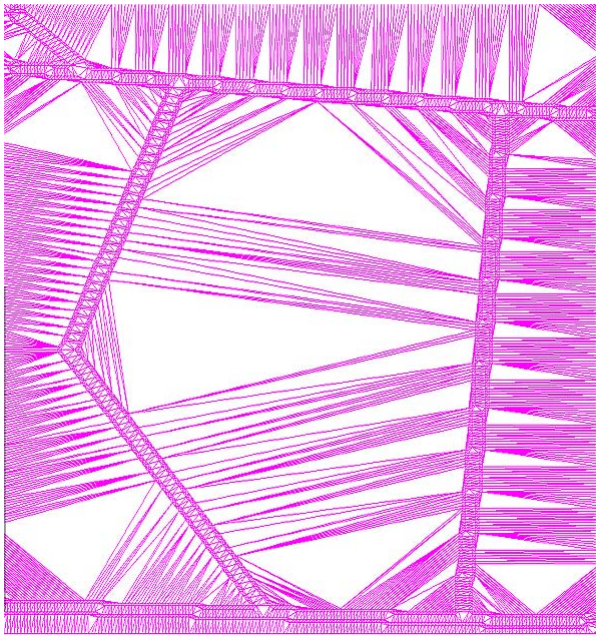

Figure 3.14: Delaunay Triangulation

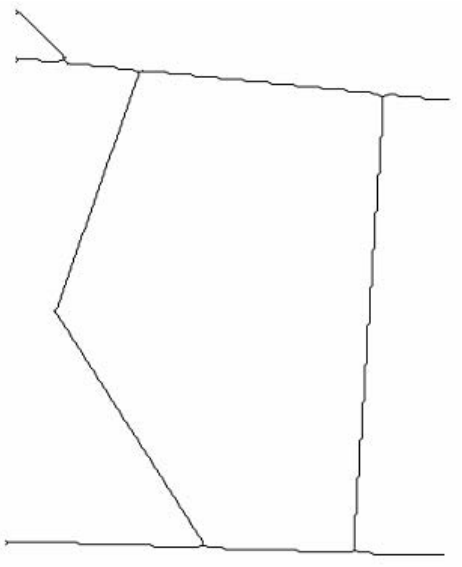

Figure 3.15: Skeleton using Voronoi Diagram

3.3.4 Skeletonization of Grayscale Maps by Voronoi Diagram

Using Voronoi Diagrams, centerlines of grayscale maps can be extracted in two ways, they are:

1. Converting the grayscale map into binary map (black and white map) by thresholding and then applying the above mentioned method

2. Applying the method directly on a grayscale map since we are generating the skeletons by Voronoi Diagrams using the change in the level of gray

The result of thresholding a grayscale map, converting it to a binary image and then applying the algorithm is shown in figure 3.19. Figure 3.16 is the thresholded grayscale map, figure 3.17 is the corresponding Voronoi Diagram, figure 3.18 is the corresponding Delaunay Triangulation and figure 3.19 is the Skeleton and boarder set of the Map. The program used to generate the skeleton was coded by Dr. Darka

Mioc (Assistant professor, Department of Geomatics, University of Calgary)and Dr. Francois Anton (Post Doctoral Fellow, Depatment of Computer Science, University of Calgary).

In some cases pruning was not efficient so we had to desin a rigorous pruning algorithm. A screenshot of the output of skeletonization by Voronoi diagram was taken and the crust was removed from it. Then the image was converted to a binary image and pruned in MATLAB. The pruning criteria are as folows:

- Connected objects that have fewer than a specified number of pixels are removed
- Pixel by pixel erosion until it reaches a connectivity number of 2
- This process was limited from 5 to 10 iterations depending on the length of the hair

Figure 3.16: Thresholded Grayscale Map

In the second method median filter is first applied to the original grayscale map and then the skeleton is generated using Voronoi Diagrams. Figure 3.8 is the

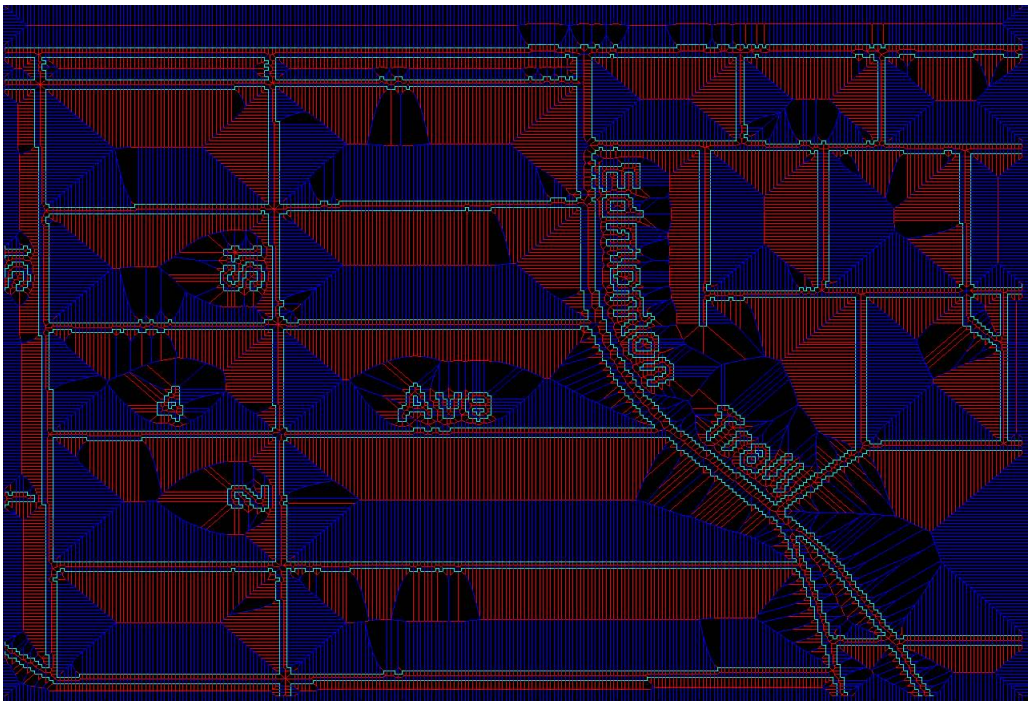

Figure 3.17: Voronoi Diagram of Thresholded Grayscale Map

original image, figure 3.20 is the corresponing Voronoi Diagram, figure 3.21 is the corresponding Delaunay Triangulation and figure 3.22 is the Skeleton with the crust. In figure 3.22 we can see that most of the lines are not pruned. This is because we had a crietria for black and white images that the lines belonging to the white region (background) was eliminated, but in a grayscale map this criteria is not valid.

3.4 Applications

Some of the applications that would benefit from this research are:

1. Geospatial Information Systems

2. Remote Sensing

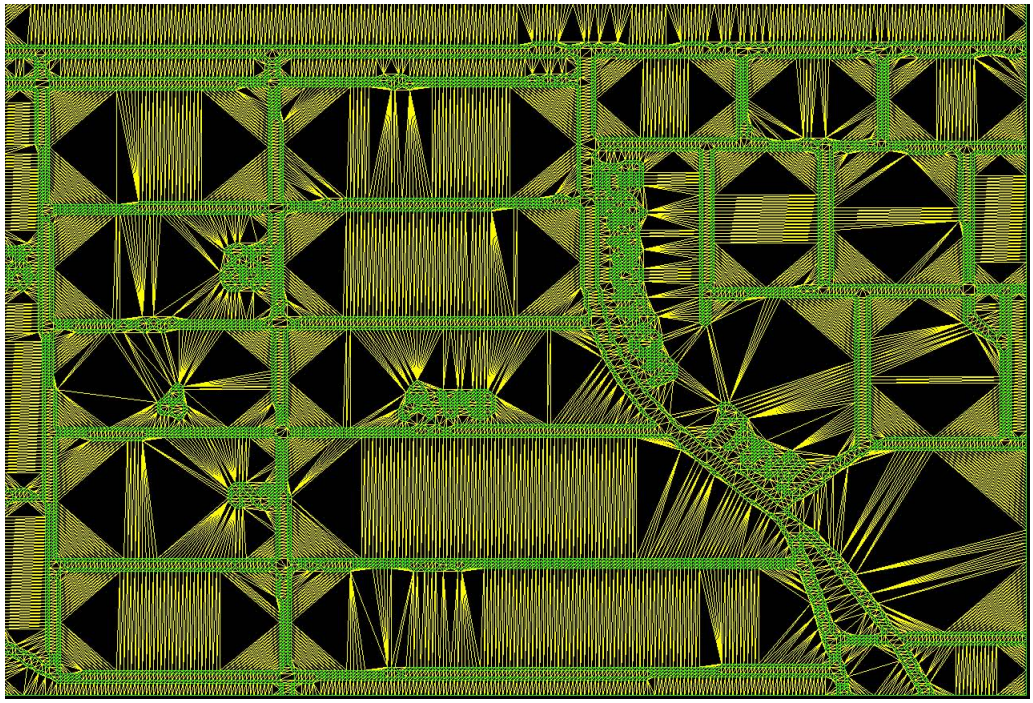

Figure 3.18: Delaunay Triangulation of Thresholded Grayscale Map

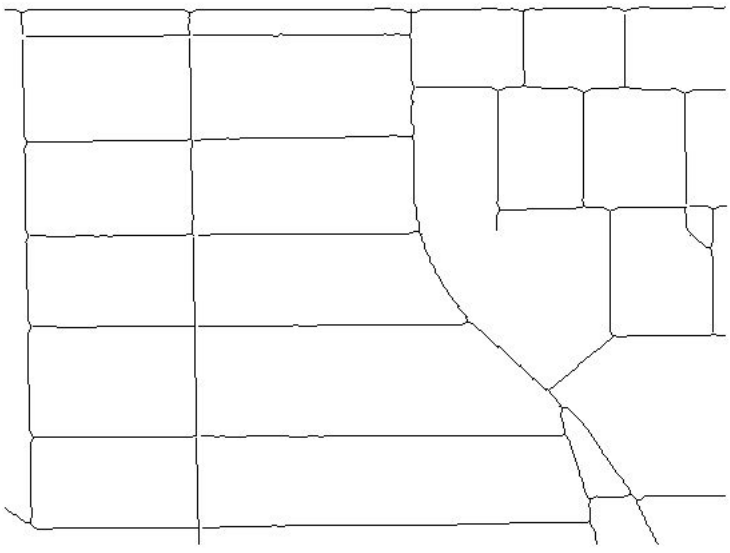

Figure 3.19: Skeleton using Voronoi Diagram

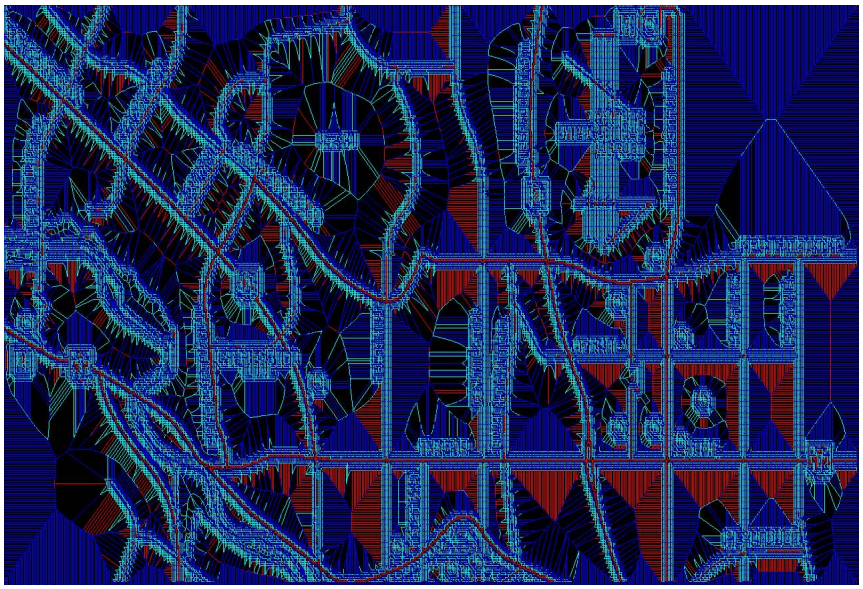

Figure 3.20: Voronoi Diagram of Grayscale Map

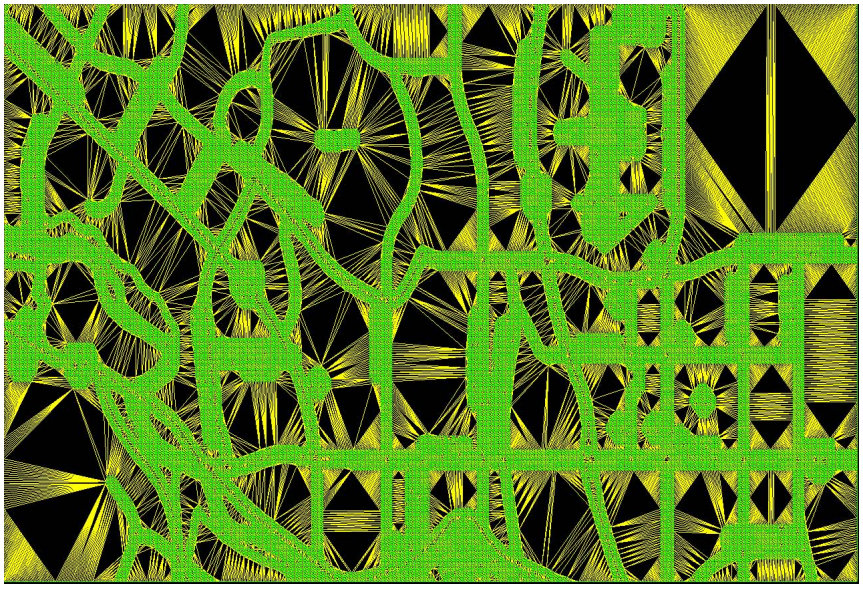

Figure 3.21: Delaunay Triangulation of Grayscale Map

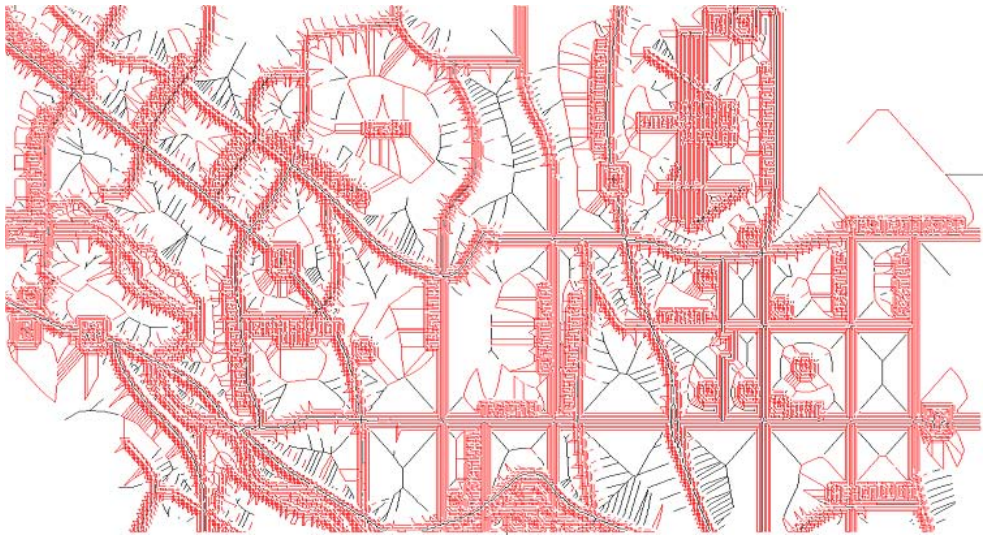

Figure 3.22: Skeleton and Crust of Grayscale Map

3. Medicine

4. Robotics

Geospatial Information Systems

This research can benifit GIS by providing an automated system of converting the available geospatial data in papaer form to digital form. However, the user has to check for accuracy and add attributes. This system can also help in updating maps etc.

Remote Sensing

This research can also help in linear feature extraction in Satellite Imageries like roads, rivers, coastal boundaries etc. It can also help in Change detection for example, if we have two satellime imageries viz., one from 1956 and the other from 2004. We can skeletonize both the images and substract the old image from the new image to get the changes. This process can also help in updating maps.

Medicine

This research can help in the field of medical sciences by skeletonizing colon, arteries, intestine, etc. which would result in the centerline. Forexample, centerline extraction of the intestine can help navigate a camera or a medical instrument through the intestine without rupturing or eroding the walls of the intestine.

Robotics

This research can also help in developing steering algorithms for robots. It can help the robot to pass through the centerline and not collide with any obstacle on its way to the destination.

We have chosen to show the application of this research in the field of remote sensing. In the next section a description of how linear features can be extracted from satellite imageries are shown.

3.4.1 Linear feature extraction from satellite imageries

The extraction of roads from satellite imageries has drawn considerable attention in recent years. The major problem is the complex structure of the image, which contains many different objects, such as roads, houses, trees, etc. with differences in shape, tone, and texture. In this section we have used small portions of grayscale satellite imageries. The methodology is explained in the next subsection.

Methodology

The flowchart of the methodology is shown in figure 3.23. The methodology is split into the following three steps:

1. Preprocessing

2. Thinning

3. Post processing

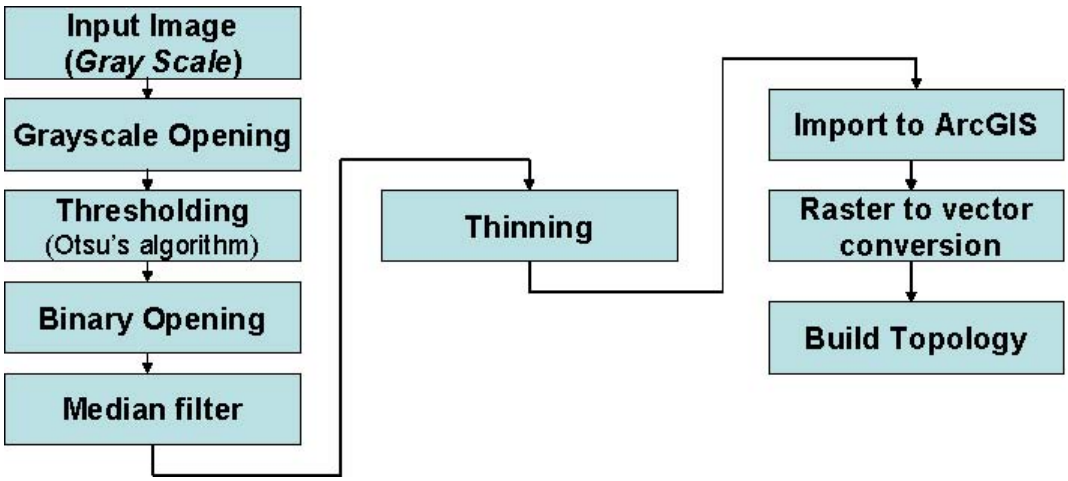

Figure 3.23: Flowchart of methodology for skeletonizing satellite image

Prepocessing

First grayscale opening is performed on the image. Opening a grayscale image is done in the same way as is done for a binary image, except that grayscale erosion and grayscale dilation operations are used; that is grayscale opening is grayscale erosion followed by a grayscale dilation using the same structuring element [42]. Grayscale dilation is a convolution-like operation. A typically small structuring element is scanned over an image and, at each position, the maximum of point-by-point sums of the image and the structuring element is computed and the maximum of each set of value is returned [42]. Similarly for grayscale erosion the minimum of point-by-pint sum of the image and the structuring element is computed and the minimum of each set of value is returned [42]. An example of grayscale opening is shown in figure 3.25 where the original image is shown in figure 3.24.

Figure 3.24: Original Satellite Image (Source: Dr. Habib)

Figure 3.25: Grayscale Opening

Then thresholding is done using Otsu's algorithm which results in a binary image. Then binary opening is done on this image to remove small isolated objects and then median filter is applied before thinning. Also, connected objects that have fewer than a specified number of pixels are removed. The result of the preprocessed image is shown in figure 3.26.

Figure 3.26: Preprocessed binary image

Thinning

Thinning is done using skeletonization by Mathematical Morphology and Voronoi Diagrams as shown in the previous sections. The result of skeletonization by Mathematical Morphology and Voronoi Diagrams are shown in figures 3.27 and 3.28 respectively.

Postprocessing

The skeltonized raster image obtained by using Mathematical Morphology is imported into ArcGIS. Then it is converted to vector data and the topology is built. The result obtained by skeletonization by Voronoi Diagram is in vector data and the topology is also built using the Quad edge data structure as shown in the previous chapter hence, the two steps of raster to vector conversion and building the topology

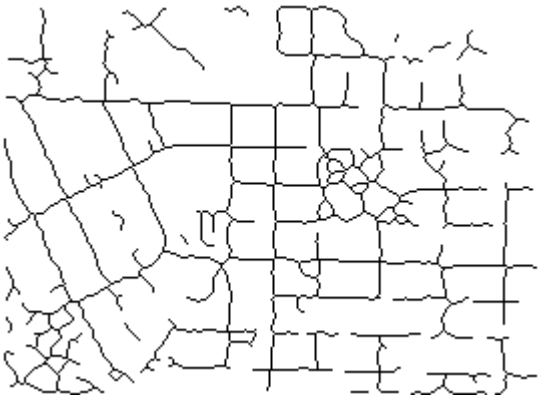

Figure 3.27: Skeletonization by Mathematical Morphology

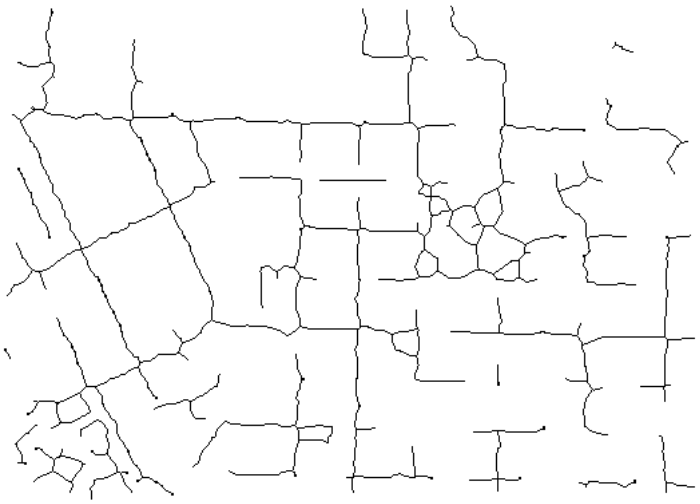

Figure 3.28: Skeletonization by Voronoi Diagrams

is not required here. Finally, after the results are imported into ArcGIS they are projected with respect to the coordinate system of the Original satellite image.

Results

The result of extracting linear features on two other grayscale satellite images are shown below. Figure 3.29 is the original image and its corresponding results are shown in figure 3.30 and figure 3.31. Figure 3.32 is the second image and its corresponding results are shown in figure 3.33 and figure 3.34.

Figure 3.29: Satellite Image

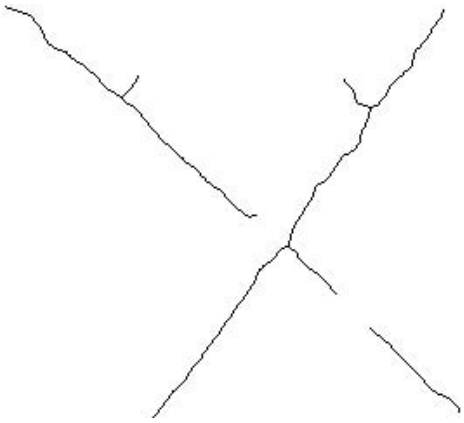

Figure 3.30: Skeletonization by Mathematical Morphology

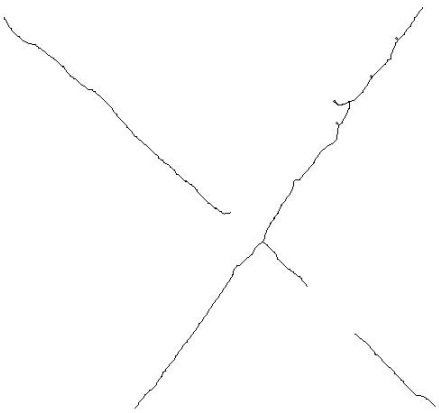

Figure 3.31: Skeletonization by Voronoi Diagrams

Figure 3.32: Satellite image

Figure 3.33: Skeletonization by Mathematical Morphology

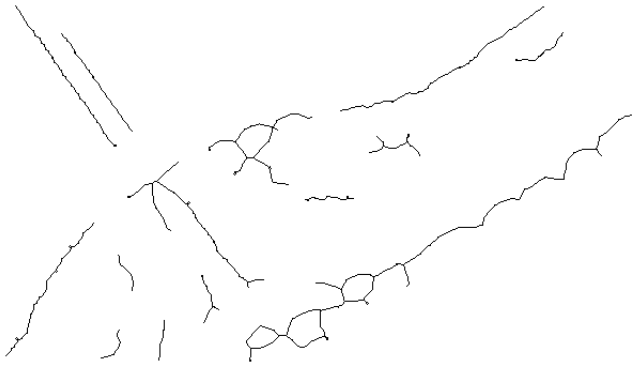

Figure 3.34: Skeletonization by Voronoi Diagrams

3.5 Summary

This chapter shows the methodology adopted to skeletonize scanned maps using Mathematical Morphology and Voronoi Diagrams. It also shows some applications of this research and the methodology of how some useful information like roads, rivers etc. can be extracted from satellite imagery using Mathematical Morphology and Voronoi Diagrams.

The next chapter shows how gap filling can be done on the skeletonized image using Least Square Parabola (LSP).

Chapter 4

Gap Filling

4.1 Introduction

In the traditional method of making maps, the surveyors survey the land and store information in the form of field notes and field drawings. This information is passed to the cartographers. Using this information the map is then drafted by the cartographer and required information is written on the drafted map. When the cartographer writes these information they erase a part of the line and write information in between them or they write the information along the line or inside a polygon etc.. For example, in a contour map the contour lines are split and the different contour levels are written in the gap. This causes problems in automatic vectorization of maps. Since the text are erased and not taken into account while vectorizing, the final output has gaps in between lines. Gaps are also produced due to noise. These gaps have to be filled in order to build the topology and do any kind of analysis like creating a buffer, intersection etc using a software on the digital map. Thus gap filling should be given prime importance after extracting the centerlines of the maps.

Therefore, after skeletonization the most important task is to reconnect the line that are disconnected by other information layers like symbols, grids etc [5]. We first skeletonize the lines and then try to find relevant extremities of the skeleton for recovering the missing parts of the lines. There are problems with this approach because the skeletonization is very sensitive to noise and usually creates small branches

at the extremities of the lines. These can be easily removed by pruning. However, by doing so, the resulting lines are shorter than the original ones and therefore some relevant information is lost. Soille **et.al** [5] have proposed a method where they are using a combination of the euclidean distance between the endpoints and their difference between their directions to join the disconnected lines.

In this research Least Square Parabola (LSP) fitting is used to fill gaps between lines. The idea behind this concept is that if two line are disconnected then a curve must pass through these lines. The best way to connect it is by identifying its end points and interpolating the points in between the end points that lie on the curve. One way of finding the curve is by using the Least square parabola fitting. In case of a line the parabola extends itself to a line.

The next section gives a brief description of LSP. Then the methodology used to connect the lines using the LSP is described. Finally the results and conclusions are given.

4.2 Least Square Parabola

The least-squares parabola uses a second degree curve as represented by the following equation [62] and [37],

$$y = a + bx + cx^2 \tag{4.1}$$

This equation is used to approximate the given set of data viz., $(x_1, y_1), (x_2, y_2), ...$ $, (x_n, y_n)$,where $n \geq 3$. The best fitting curve $f(x)$ has the least square error, i.e. [62],

$$\Pi = \sum_{i=1}^{n}[y_i - f(x_i)]^2 = \sum_{i=1}^{n}[y_i - (a + bx_i + c(x_i)^2)]^2 = \min \quad (4.2)$$

Note that a, b and c are unknown coefficients while all x_i and y_i are given. To obtain the least square error, the unknown coefficients a, b and c must yield zero first derivatives. They are shown in the following equation,

$$\frac{\partial \Pi}{\partial a} = 2\sum_{i=1}^{n}[y_i - (a + bx_i + c(x_i)^2)] = 0 \quad (4.3)$$

$$\frac{\partial \Pi}{\partial b} = 2\sum_{i=1}^{n} x_i[y_i - (a + bx_i + c(x_i)^2)] = 0 \quad (4.4)$$

$$\frac{\partial \Pi}{\partial c} = 2\sum_{i=1}^{n} x_i^2[y_i - (a + bx_i + c(x_i)^2)] = 0 \quad (4.5)$$

Expanding the above equations, we have

$$\sum_{i=1}^{n} y_i = a\sum_{i=1}^{n} 1 + b\sum_{i=1}^{n} x_i + c\sum_{i=1}^{n} x_i^2 \quad (4.6)$$

$$\sum_{i=1}^{n} x_i y_i = a\sum_{i=1}^{n} x_i + b\sum_{i=1}^{n} x_i^2 + c\sum_{i=1}^{n} x_i^3 \quad (4.7)$$

$$\sum_{i=1}^{n} x_i^2 y_i = a\sum_{i=1}^{n} x_i^2 + b\sum_{i=1}^{n} x_i^3 + c\sum_{i=1}^{n} x_i^4 \quad (4.8)$$

The unknown coefficients a, b and c can be obtained by solving the above linear equation [62].

Thus this method can be used to approximate the given set of data x and y to find the three unknown coefficients viz., a, b and c that represent the curve [62].

4.3 Methodology

The flowchart of the methodology is shown in figure 4.1. The algorithm is split into the following 10 steps:

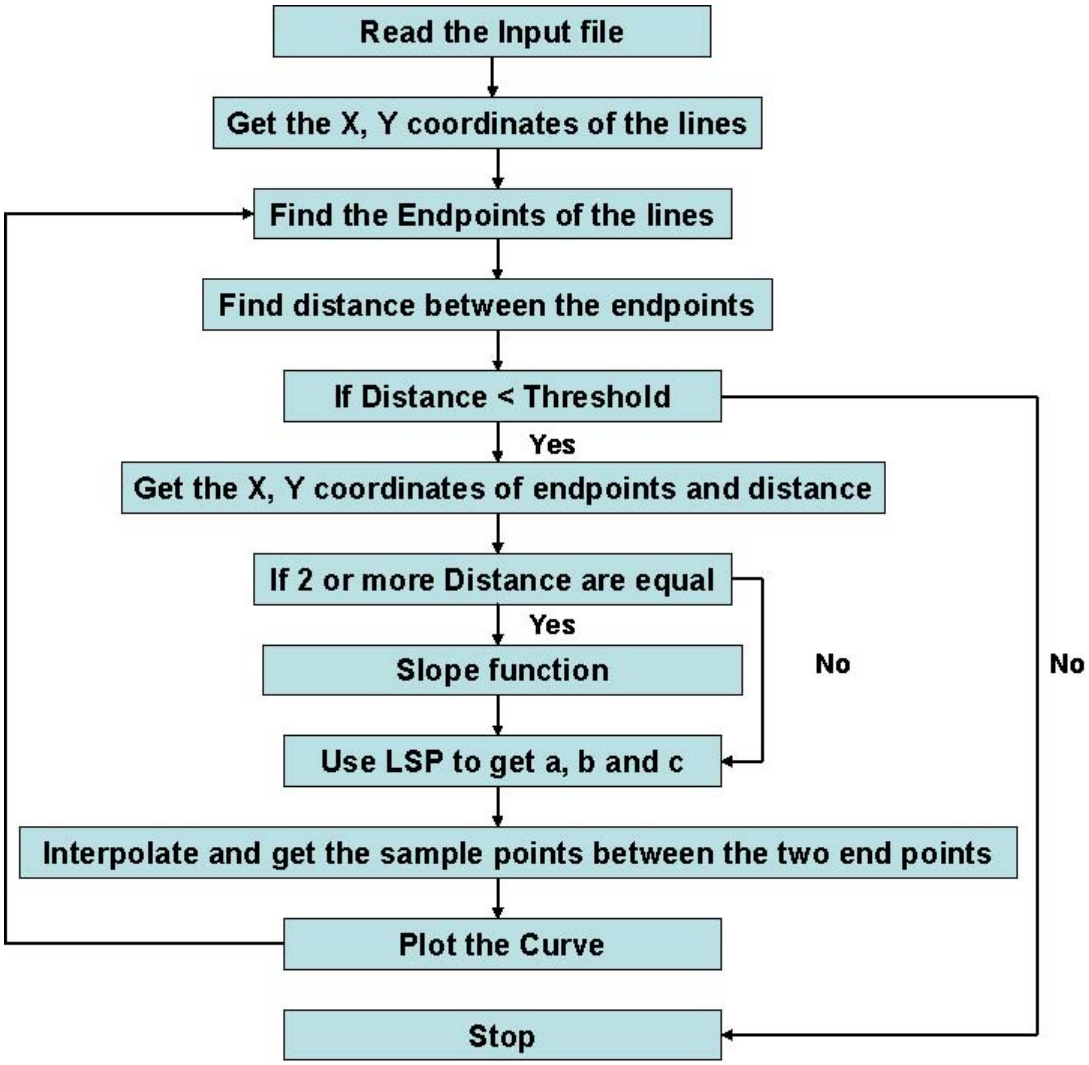

Figure 4.1: Flowchart for Gap Filling

Step 1: Reading the input and getting the x and y coordinates of the line

The input for this method is a one pixel width binary image that can be obtained as a result of skeletonization that is explained in the previous chapter. Consider a black and white image with the object of interest black in color. The methodology involves reading the image first to identify the object pixels. Then to extract the x coordinate and y coordinate of the object pixels. Since we want to fill the gap and not approximate the original line, we interpolate and draw only the gap and connect it with the end points. In order to do this we have to find the end points.

Step2: Get x and y coordinates of the endpoints

In order to get the x and y coordinate of the end points we have to identify the endpoints and get the x and y coordinates of those points. A point is an endpoint if its connectivity number is 1.

Step 3: Find distance between endpoints

After finding the end points we find the distance between the end points using the Euclidean distance formula which can be mathematically represented as,

$$D = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{4.9}$$

where D is the Euclidean distance and (x_1, y_1) and (x_2, y_2) are endpoints.

Step5: Setting the threshold

Now we put a condition that if the distance is less than a threshold value then these two line will be considered for gap filling otherwise they will not be considered and the program will stop. This condition is necessary because if we consider a case where there are two line to be connected. We have four endpoints for these 2 lines. If we do not set a threshold value then the gapfilling algorithm will connect all the endpoints with each other, so it will result in six lines. However, in this case we also

assume that the gap is smaller than the disconnected line segments. The value of the threshold depends on the longest gap between two lines that are disconnected in the image. In this research we have chosen a threshold of 20 pixels. In this program the user has an option to set the threshold. If the threshold is not specified then by default it is taken as 20 pixels.

Step6: Get the x and y coordinate of end points and five adjacent points coresponding to the line

Then we get the x and y coordinate of the end points that have distance that is less than the set threshold. Now, we find the coordinates of the lines that correspond to these end points and get the x and y coordinates of five adjacent pixels along the same line including the endpoints. For example, consider two lines that are to be connected with x and y coordinates (1, 1), (2, 2)...(7, 7) and (11, 11), (12, 12)....(20, 20) for the first and second line respectively. The endpoints that are to be connected are (7, 7) and (11, 11). Therefore the x and y coordinates of the five adjacent pixels along the same line including the endpoints are: (3, 3), (4, 4), (5, 5), (6, 6) and (7, 7) and for the second line are: (11, 11), (12, 12), (13, 13), (14, 14) and (15, 15).

Step7: Check if any of the distance are equal

After getting the list of x and y coordinate of the end points, their distance and the corresponding five adjacent points of each line we have to check if any of the distance match each other. If any two or more distances match then we again check if any of the x and y coordinate of the end points of the lines are common. If it is common then we know that there is a competition between three or more lines. If there is a competition then we go to step 8 (slope function) otherwise go to step 9 (Least Square Parabola).

Step8: Slope Function

In this function we calculate the slope of the lines that have equal distance. If there are three lines to be intersected and they are at equal distance from each other then we calculate the slope of all the lines. Slope of a horizontal line is zero, a vertical line is infinity and a line at a 45° inclination is 1. If any line has a slope of zero or infinity then we connect the other two lines with this line. If the slope of two lines have the same magnitude but opposite directions then we connect these two lines first and then we compute the difference between the slope of the third line with the other two line. Then we connect the lines that have minimum difference between their slopes.

For cases where none of the three lines have slope of zero or infinity we get the difference between the slopes of all the lines and connect the lines with minimum difference between the slopes.

Step9: Least Square Parabola fitting and other cases

Using these coordinates we perform least square parabola fitting to get the values of the coefficient, a, b and c. Using the values of a, b and c and the x coordinates of the two lines we can get an approximate value of y. There are other cases where we can directly extend the line and we do not have to approximate the curve. The X and Y coordinate are chosen based on four cases as shown below. Let us consider that (x_1, y_1) and $(x_2, y2)$ are the end points of two lines whose distance is less than the threshold value.

Case 1

Consider figure 4.2, the end points are highlighted in red. Here we can see that $x_1 = x_2$. If this is the case then x is a constant and (x_1, y_1) and (x_2, y_2) must be connected using a straight line. So the minimum value of x is incremented by 1 until it reaches the maximum value of x. For example, in figure 4.2 x is a constant and $x_1 = x_2 = 5$. $y_1 = 3$ and $y_2 = 8$ now we increment minimum value of y by 1 until it reaches the maximum value of y (y_1 by 1 until it reaches y_2) so the values of y will be, 4, 5, 6 and 7. So the corresponding coordinates will be, $(5, 4), (5, 5), (5, 6)$ and $(5, 7)$. An example of gapfilling of this case is shown in figure 4.3.

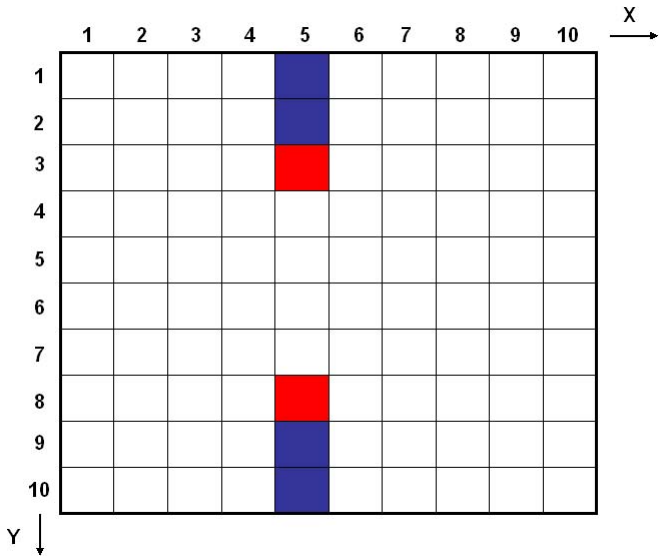

Figure 4.2: Case 1: Gap Filling

Case 2

Consider figure 4.4, the end points are highlighted in red. Here we can see that $y_1 = y_2$. If this is the case then y is a constant and (x_1, y_1) and (x_2, y_2) must be connected using a straight line. So minimum value of x is incremented by 1 until

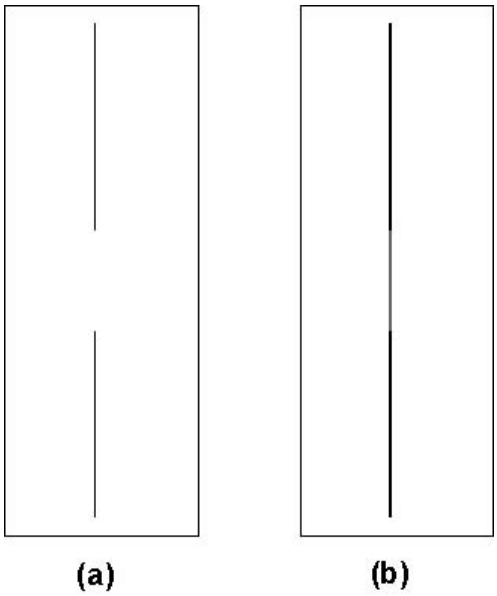

Figure 4.3: Example of Gapfilling (case 1)

it reaches the maximum value of x. For example, in figure 4.4 y is a constant and $y_1 = y_2 = 4$. $x_1 = 3$ and $x_2 = 9$ now we increment x_1 by 1 until it reaches x_2 so the values of x will be, 4, 5, 6, 7 and 8. So the corresponding coordinates will be, $(4,4), (4,5), (4,6)(4,7)$ and $(4,8)$. An example of gapfilling of this case is shown in figure 4.5.

Case 3

Consider figure 4.6, the end points are highlighted in red. Here we can see that $x_1 = y_1$ and $x_2 = y_2$. If this is the case then $x = y$ and (x_1, y_1) and (x_2, y_2) must be connected using a straight line. So minimum value of x is incremented by 1 until it reaches maximum value of x and simultaneously minimum value of y is incremented by 1 until it reaches maximum value of y. For example, in figure 4.6 $x_1 = y_1 = 3$ and $x_2 = y_2 = 7$. Since $x_1 = y_1 = 3$ and $x_2 = y_2 = 7$ now we increment x_1 and

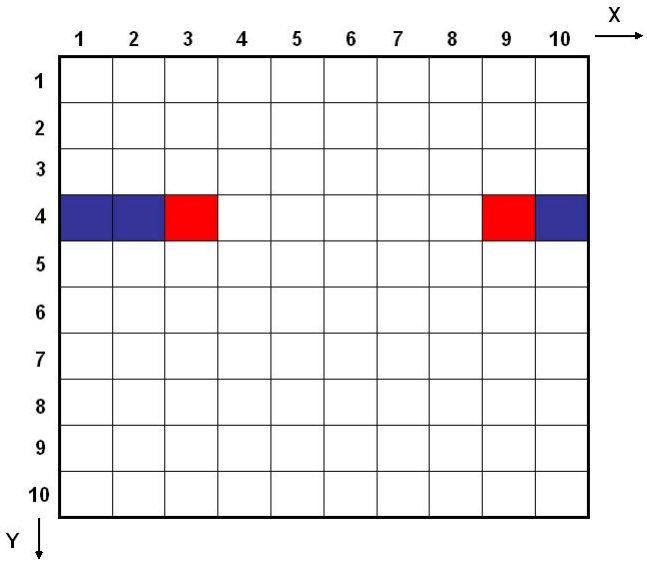

Figure 4.4: Case 2: Gap Filling

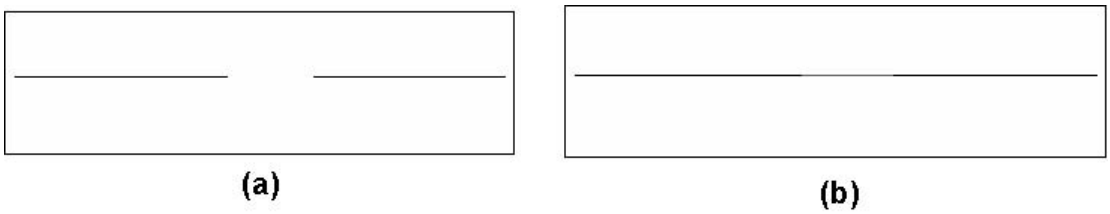

Figure 4.5: Example of Gapfilling (case 2)

$y_1$by 1 until it reaches x_2 or $y_2$so the values of x and y will be, 4, 5 and 6. So the corresponding coordinates will be, $(4,4),(5,5)$ and $(6,6)$. An example of gapfilling of this case is shown in figure 4.7.

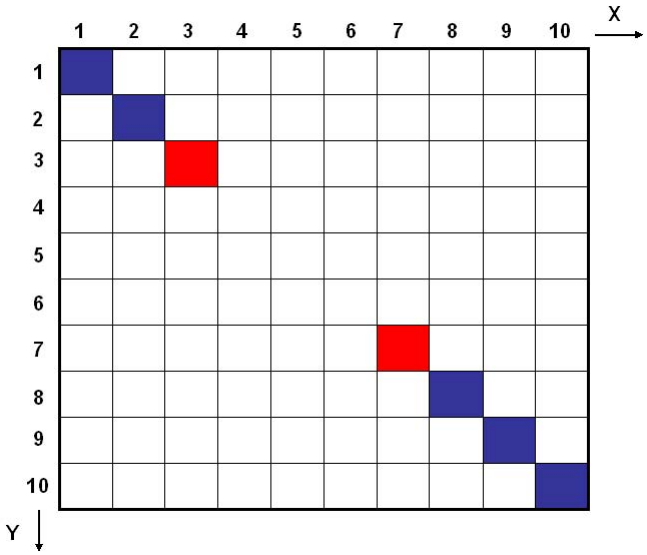

Figure 4.6: Case 3: Gap Filling

Case 4

Consider figure 4.8, the end points are highlighted in red. Here we can see that $x_1 \neq x_2$ and $y_1 \neq y_2$ and $x_1 \neq y_1$ and $x_2 \neq y_2$. In this case since $x \neq y$ we cannot connect it using a straight line and so we will use Least Square parabola to interpolate the points in between the endpoints.

Using the x and y coordinates of the two lines we get the value of a, b and c using the steps explained in the above section. After we get the values of a, b and c we increment minimum value of x by 1 until it reaches maximum value of x and substitute the vale of x in the following equation to get the corresponding y value,

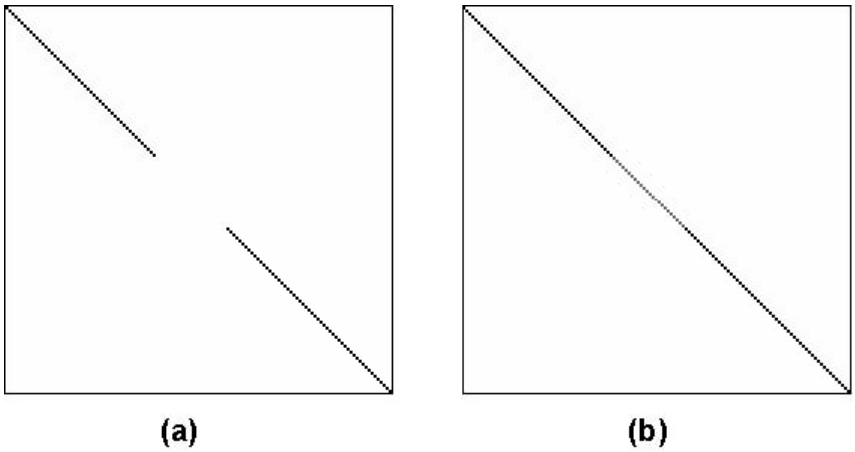

Figure 4.7: Example of Gapfilling (case 3)

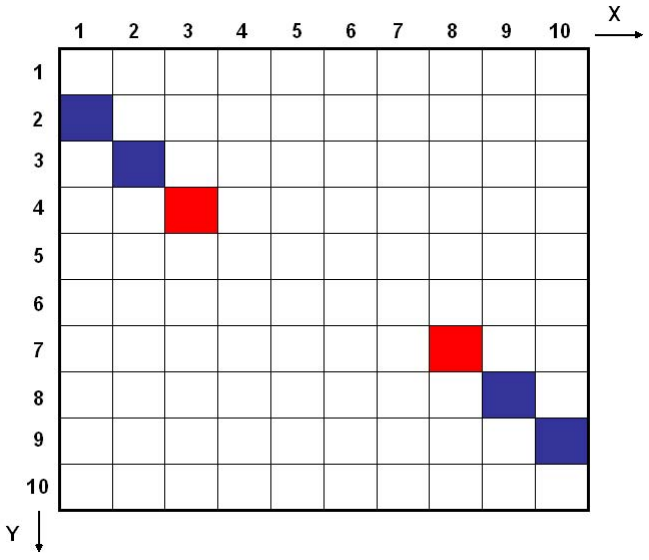

Figure 4.8: Case 4: Gap Filling

$$f(x) = a + bx + cx^2 \tag{4.10}$$

Where $f(x) = y$. The least squares line method uses this equation to get the parabola graph. After getting the value of y we approximate the number to a natural number. The condition for approximation being that if the decimal value is greater than or equal to 0.5 then it is approximated (rounded) to the next number and if it is less than 5 then it is approximated to the real number. For example if the value of y is 4.75 then it is approximated to 5 and if the value of $y = 4.30$ then it is approximated to 4. An example of gapfilling of this case is shown in figure 4.9.

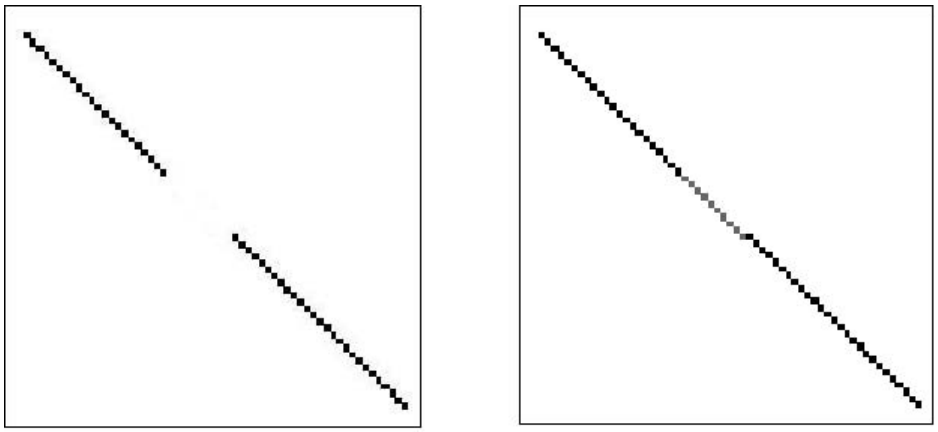

Figure 4.9: Example of Gapfilling (case 4)

Rounding the numer or approximating is only done for raster images and not for vector data since there is no need to rasterise the curve. LSP is used only for case four because in the other cases we get the exact coordinates by just extending the line and we do not have to interpolate and get the x coordnate to get the corresponding y coordiante.

Case5

Consider figure 4.10, the end points are highlighted in red. Here we can see that $x_1 \neq x_2$ but the difference between them is 1. Now we find if $x_1 < x_2$ or $x_1 > x_2$. In figure 4.10 $(x_1, y_1) = (5, 3)$ and $(x_2, y_2) = (6, 8)$. Therefore $x_1 < x_2$. Now we increment the minimum value of y by 1 until it reaches the maximum value of y, i.e. we increment y_1 by 1 until it reaches y_2 therefore, the values of y will be, 4, 5, 6 and 7. So the corresponding coordinates will be, $(5, 4), (5, 5), (5, 6)$ and $(5, 7)$. An example of gapfilling of this case is shown in figure 4.11.

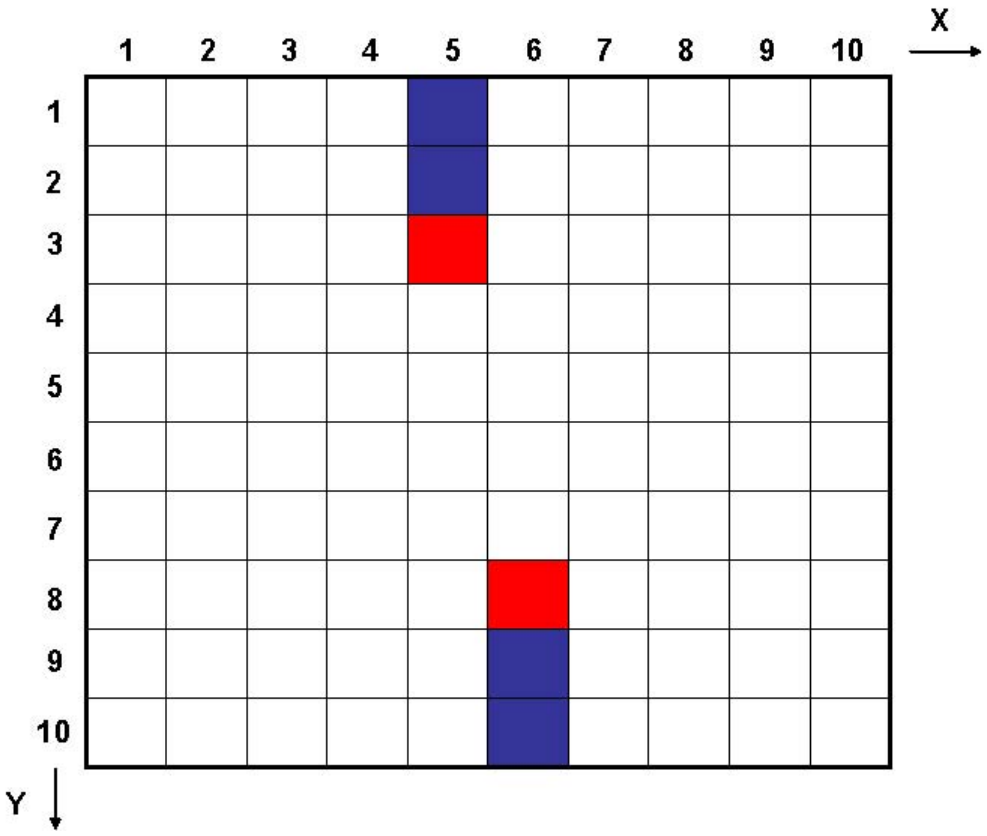

Figure 4.10: Case 5: Gapfilling

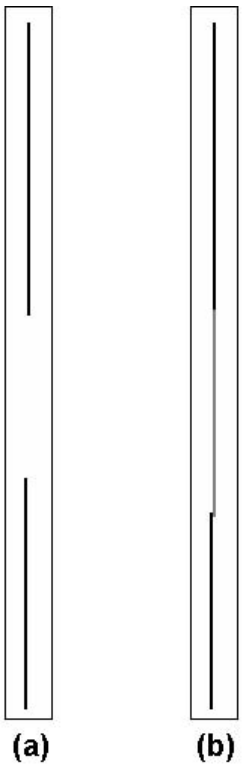

Figure 4.11: Example of Gapfilling (case 5)

Case6

Consider figure 4.12, the end points are highlighted in red. Here we can see that $y_1 \neq y_2$ but the difference between them is 1. Now we find if $y_1 < y_2$ or $y_1 > y_2$. In figure 4.12 $(x_1, y_1) = (3, 4)$ and $(x_2, y_2) = (9, 5)$. Therefore $y_1 < y_2$. Now we increment the minimum value of x by 1 until it reaches the maximum value of x, i.e. we increment x_1 by 1 until it reaches x_2 therefore, the values of x will be, 4, 5, 6, 7 and 8. So the corresponding coordinates will be, $(4, 4), (4, 5), (4, 6)(4, 7)$ and $(4, 8)$. An example of gapfilling of this case is shown in figure 4.13.

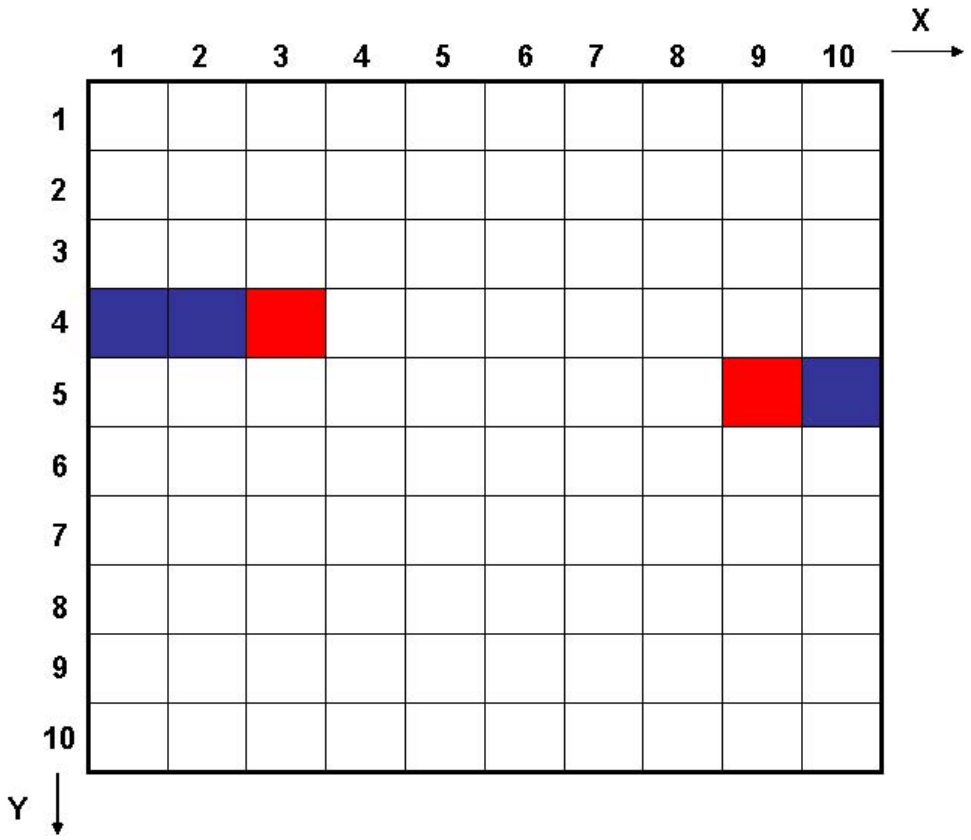

Figure 4.12: Case 6: Gapfilling

Step 10: Plot the curve and redraw the image

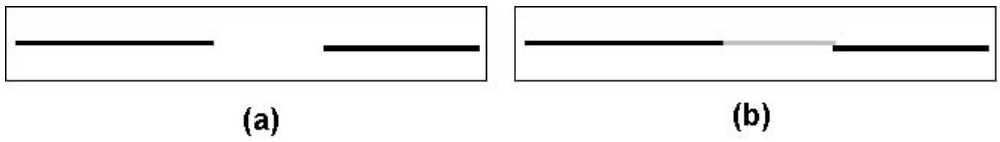

Figure 4.13: Example of Gapfilling (case 6)

Finally, we plot the curve with the coordinates obtained from step 9. When this curve is drawn it fills the gap. Now, we draw the image using the size of the original image and the x and y coordinates obtained from step 9. We give it a value of 0.8 so that it appears in gray color and we can easily spot the connection between lines. Then we add the original image with this image to get the complete gapfilled image.

Another important point that should be noted is that the lines to be connected should be labeled before gap filling. For example, roads, rivers and buildings are given a label ID of 1, 2, 3 and 4. Hence, in a raster image the program is run 4 times and each time only the gaps between the same label Id's are connected. This way we can prevent misconnections. Whereas, while gap filling in vector data the different layers can be separated and the algorithm can be applied to individual layers.

The next section shows the result obtained by using Least Square parabola fitting for gap filling.

4.4 Results

The result obtained by using Least Square parabola fitting for gap filling is shown in the following figures. Figure 4.14 and 4.15 are the original image and figure 4.16 and 4.17 are the corresponding gapfilled image.

Since this is an iterative process all the gaps that are within the threshold are

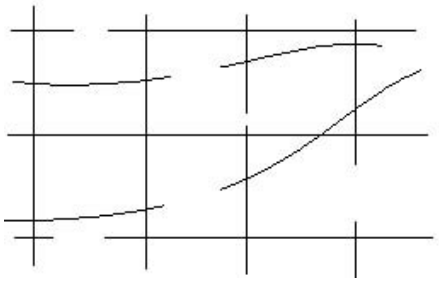

Figure 4.14: Original Image

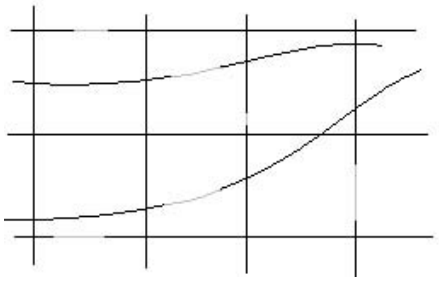

Figure 4.15: Gap Filled Image

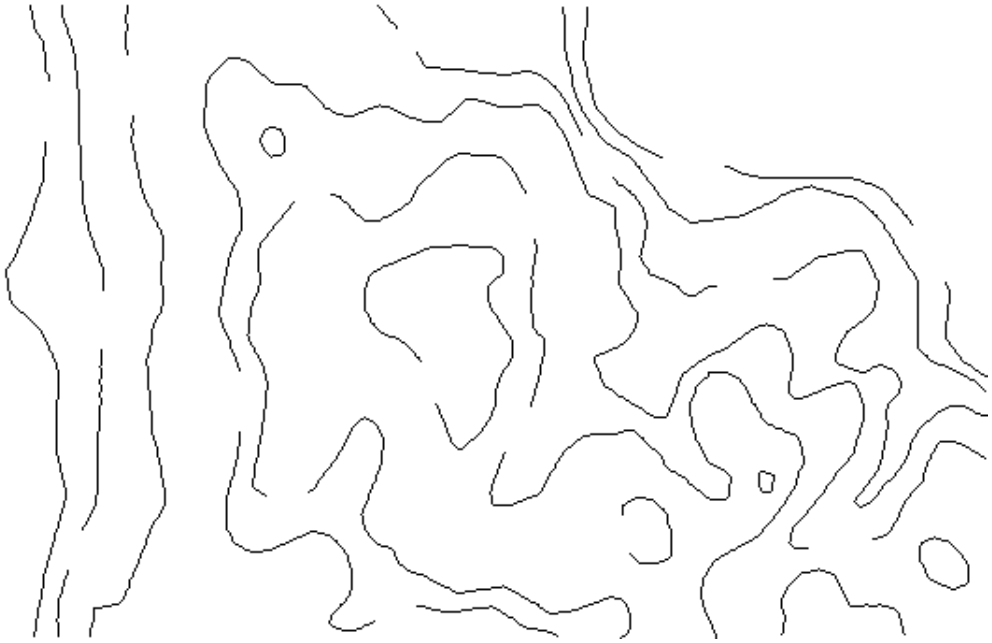

Figure 4.16: Contour Map with Gap

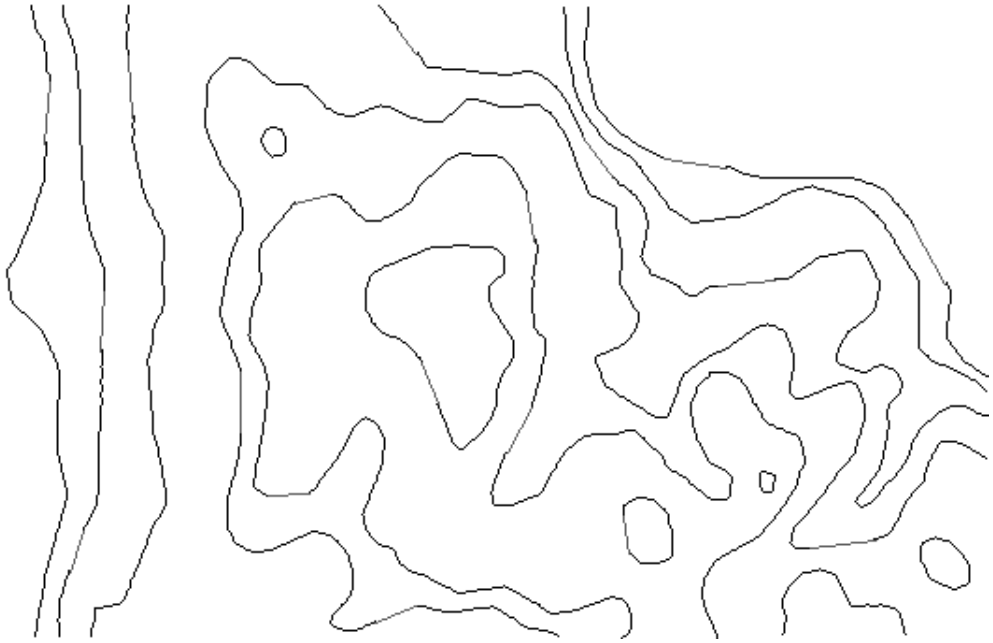

Figure 4.17: Gap Filled Contour Map

filled. Our method works well for lines and curves since we take the distance, slope and the best fitting curve into account for gap filling. It also works well for contour lines that are close to each other and prevents the lines from snapping to each other and draws the continuous curve. This method produces accurate results for vector data since we directly plot the curve between the gap and we do not have to round the values for approximation. Gap filling in raster data is basically an approximation of the computed curve as shown in case 4. The curve is first rasterized and then the respective pixels are plotted. The filled gaps are shown in gray and the original lines are shown in black color in the results.

4.5 Conclusion

There are semi automatic methods available to connect disconnected lines using line tracing algorithms [5]. We have presented an automated method for gap filling that is used to connect disconnected lines using Least Square Parabola fitting. Gap filling using LSP works better for vector lines than for raster data due to the approximation of the curve to pixels in raster data. This method can prove to be very useful in the GIS and Image processing world.

4.6 Summary

In this chapter we have seen how least square parabola fitting can be used to fill gaps and connect disconnected lines in a map.

The next chapter, Chapter 5, shows the results of skeletonization by Mathematical Morphology and Voronoi Diagrams and compares the vectorized map with respect to connectivity, the way it handles curves, topology, pruning, processing time etc.

Chapter 5

Comparison of Results

Vectorisation is the most fundamental operation in interpretation of line drawings and document analysis. Choosing a vectorization method that best suits the needs of the system is very important. In general, good methods must preserve information like line geometry and intersection junction as far as possible. It is also important to analyse the error and find the accuracy of the result with respect to the original data. We have chosen Skeletonization by Mathematical Morphology and Voronoi Diagrams for vectorizing maps.

Comparison of skeletonization using Mathematical Morphology and Voronoi Diagrams are done based on the following criteria:

- How does it handle junction and curves?
- Error Analysis

5.1 Handling of curves and junctions

The result of skeletonizing a black and white image is shown in the following figures:

Figure 5.1 is a simple black and white image. The result of skeletonization by Mathematical Morphology is shown in figure 5.2 and the result of skeletonization by Voronoi Diagrams is shown in figure 5.3. Both the programs are run on MacIntosh Apple eMAC with configuration, 1GHz PowerPC G4, 256MB, 80GB.

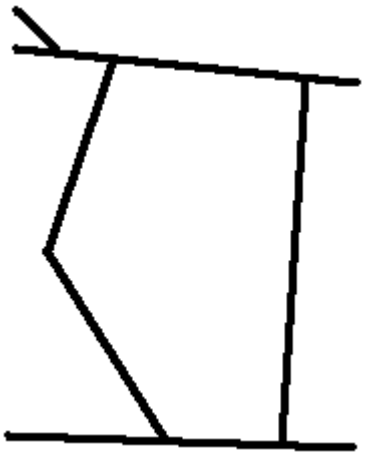

Figure 5.1: Original Image

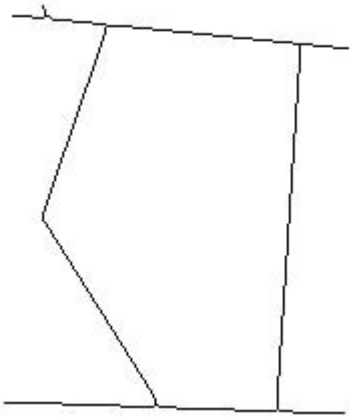

Figure 5.2: Skeletonization by Mathematical Morphology

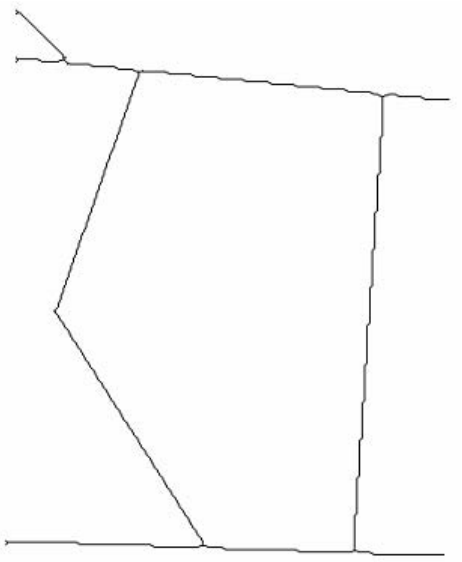

Figure 5.3: Skeletonization by Voronoi Diagrams

Considering figure 5.2 and 5.3 we can see that both the methods result in the centerline of the image. Some portions of the image are zoomed to make a clear comparison of the two methods. In figure 5.4, 5.5 and 5.6 (a) is the original image, (b) is the result of skeletonization by Mathematical Morphology and (c) is the result of skeletonization by Voronoi Diagrams. Comparison of curves is shwon in figure 5.4. Comparison of junctions is shown in figure 5.5 and comparison of corners is shown in figure 5.6

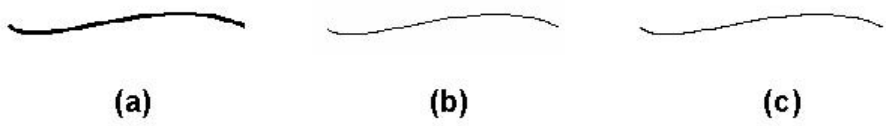

Figure 5.4: Comparison of Curves

In figure 5.4 we can see that both the methods handle curves well. In figure 5.5 we can see that Mathematical Morphology handles junctions better than Voronoi

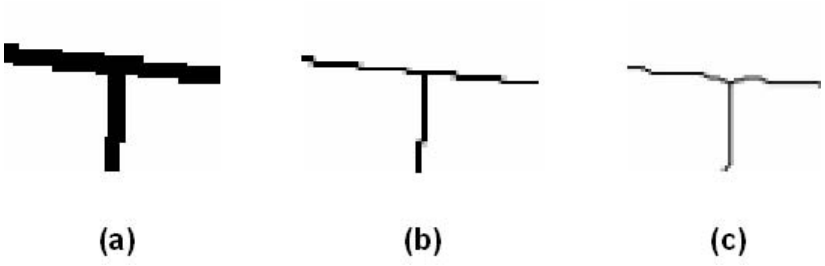

Figure 5.5: Comparison of Junction

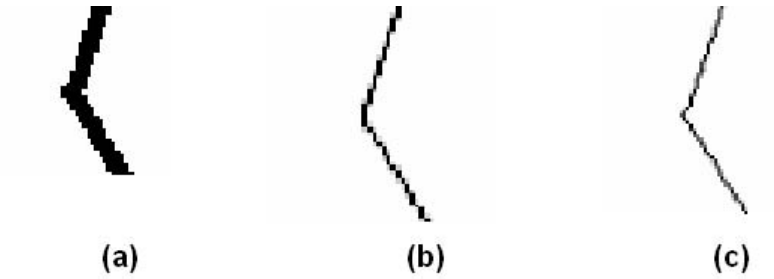

Figure 5.6: Comparison of corners

Diagrams. Mathematical Morphology preserves the geometry of the T-intersections (junctions) where as Voronoi diagram induces a small dip in the intersection. In figure 5.6 we can see that Voronoi diagram preserves the geometry of the corner where as Mathematical Morphology blunts the corner due to heavy erosion and another reason might also be the preprocessing step of removing the classical thinning artifacts.

5.2 Error Analysis

The knowledge that we have of the physical world is obtained by doing experiments and making measurements. It is important to understand how to express such data and how to analyze and draw meaningful conclusions from it [53]. We have chosen to use the mean and standard deviation to show the estimated error or accuracy of

the matched points.

In our case, we have done the first step of vectorizion, i.e. convertion of raster data to vector data using skeletonization by Mathematical Morphology and Voronoi Diagrams. The map was also digitized manually by me for comparison. Our main objective now is to find the accuracy or the deviation of the vectorized data with the original data. In order to do this we took a printout of two portions of the Calgary downtown vector data that was provided by the library of the University of Calgary (Source: The City of Calgary) and scanned it at a resolution of 300×300 dpi in the Microlab, Department of Geomatics Engineering, University of Calgary using the Hewlett-Packard (HP) Scanjet 6100C scanner. The two scanned maps are shown in figure, 5.7 and 5.11. The result of manual digitization, skeletonization using Mathematical Morphology and Voronoi Diagrams of the first map are shown in figure 5.8, 5.9 and 5.10 respectively. The result of manual digitization, skeletonization using Mathematical Morphology and Voronoi Diagrams of the second map are shown in figure 5.12, 5.13 and 5.14 respectively.

The result of skeletonization was imported into ArcGIS and was georeferenced and projected. The projection information are as follows:

3TM Projection: Transverse Mercator

False Easting: 0.000000

False Northing: 0.000000

Central Meridian: -114.000000

Scale Factor: 0.999900

Latitude Of Origin: 0.000000

Unit: Meter

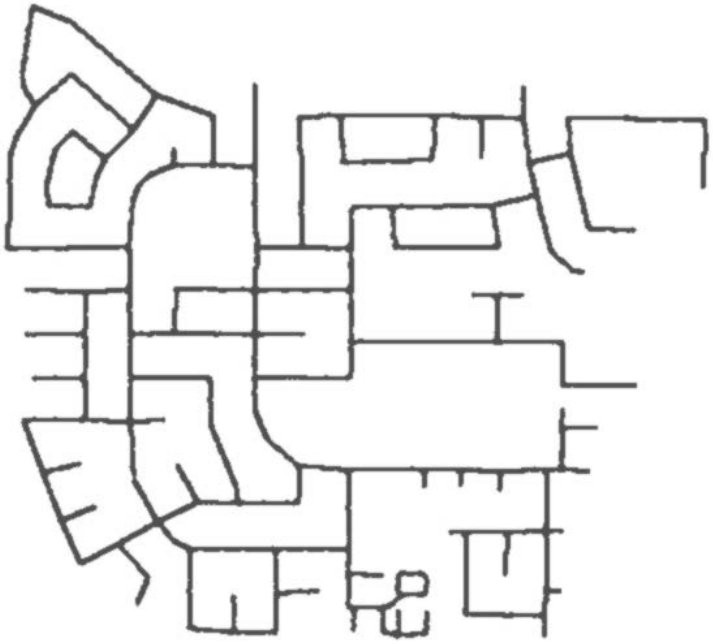

Figure 5.7: Scanned Map 1

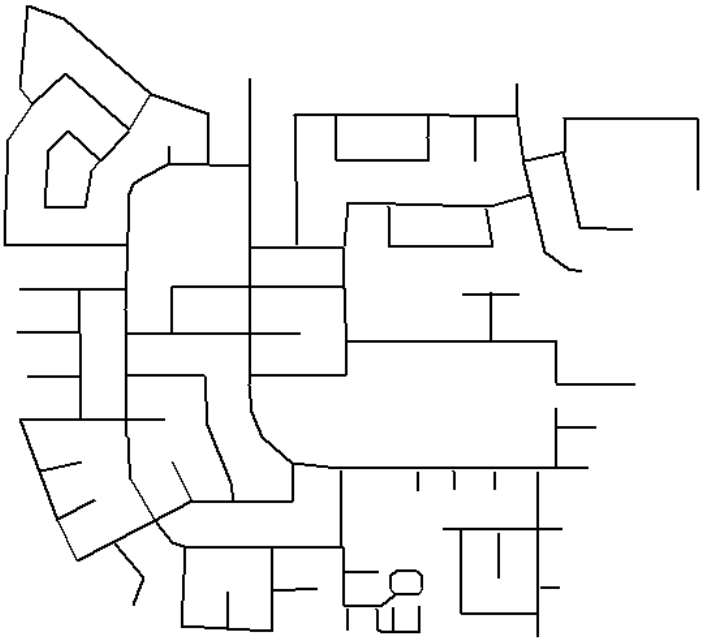

Figure 5.8: Manually Digitized (Scanned Map1)

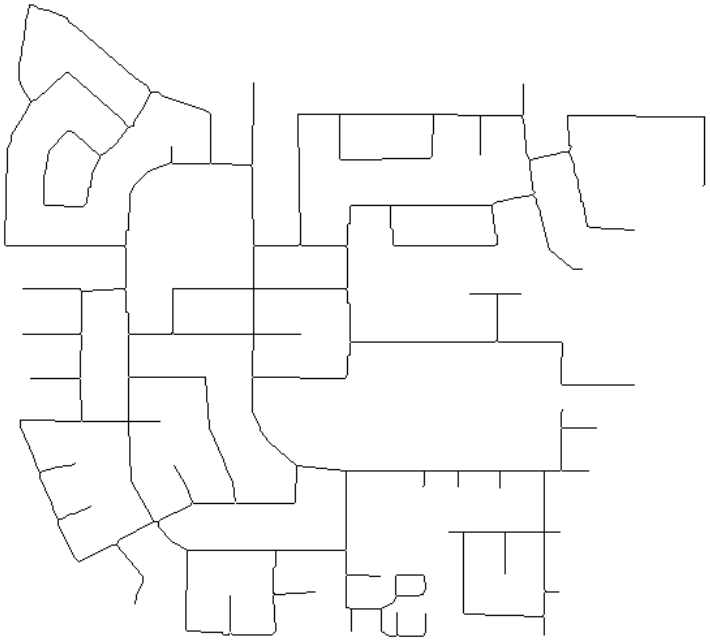

Figure 5.9: Skeletonization by Mathematical Morphology

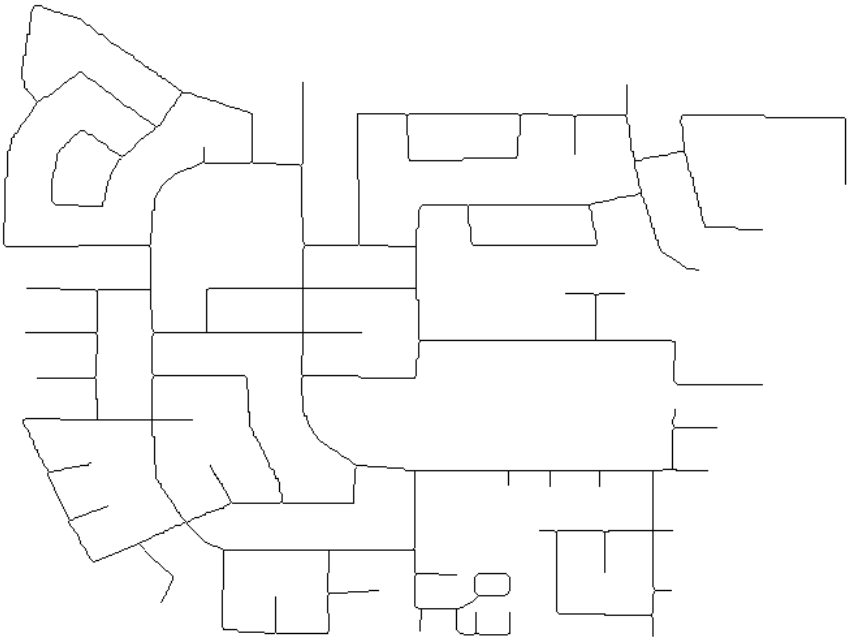

Figure 5.10: Skeletonization by Voronoi Diagrams

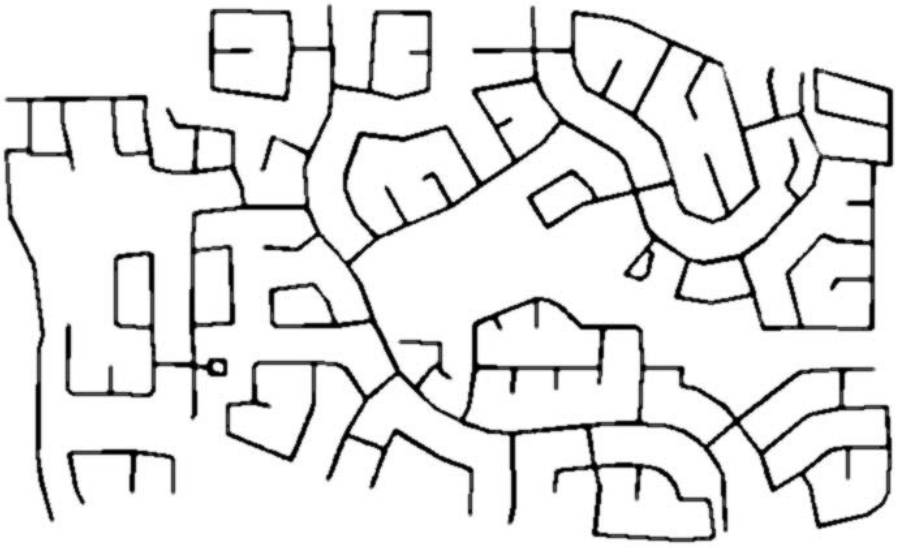

Figure 5.11: Scanned Map 2

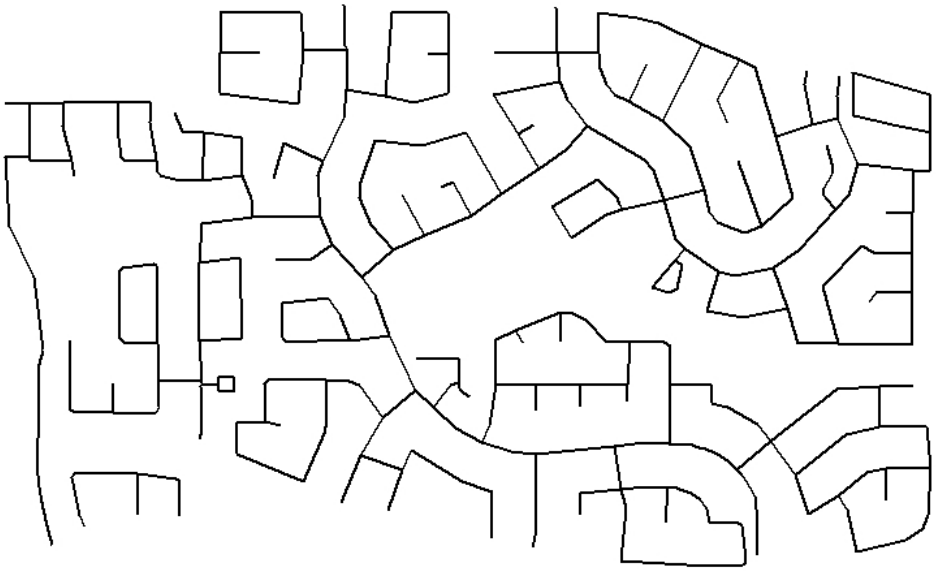

Figure 5.12: Manually Digitized (Scanned Map2)

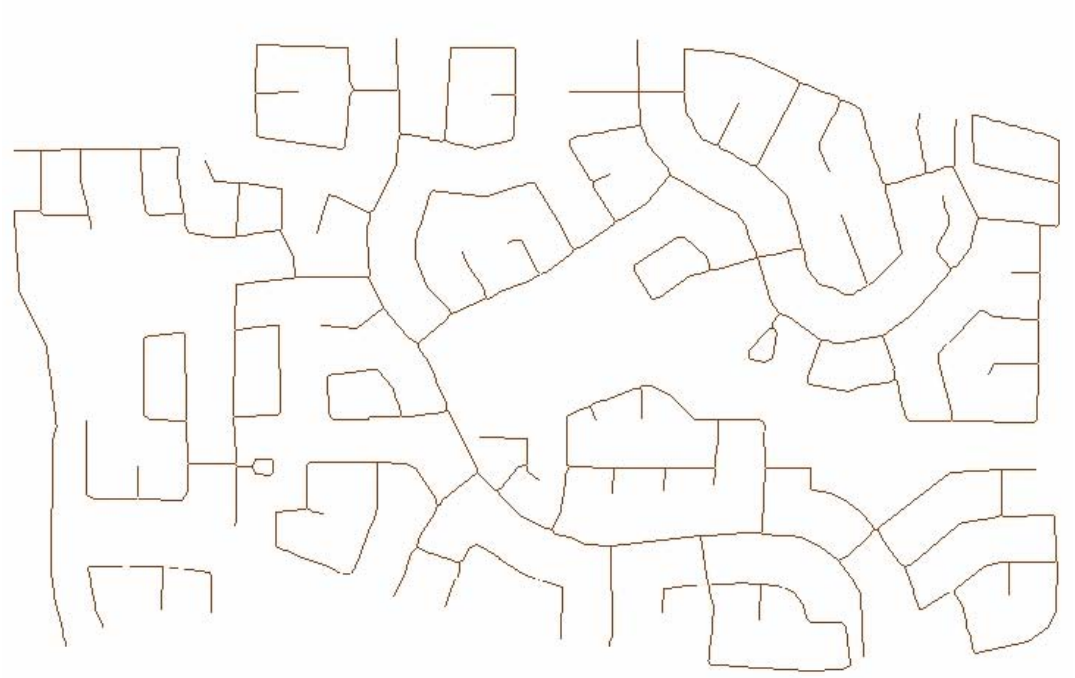

Figure 5.13: Skeletonization by Mathematical Morphology

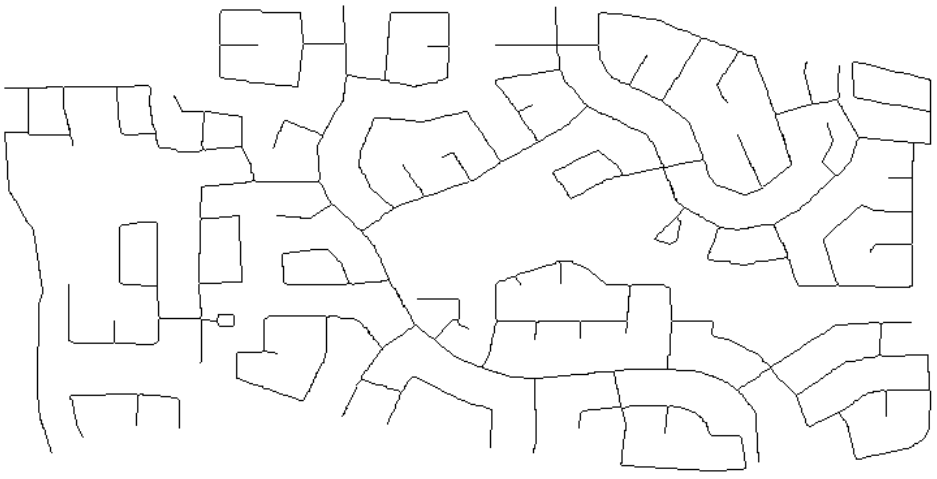

Figure 5.14: Skeletonization by Voronoi Diagrams

Then the maps were overlayed on the original data. The result of overlay of skeletonization by Mathematical Morphology over the original map is shown in figure 5.15. After overlaying the map it was found that manual digitization had an error of upto 2 meters. The result produced by Mathematical Morphology had an error of upto 3.5 meters and the result generated by Voronoi Diagrams had an error of upto 5 meters. These errors were measured using the measuring tool in ArcGIS. It was also found that the error was not continous and in most of the locations it matched completely.

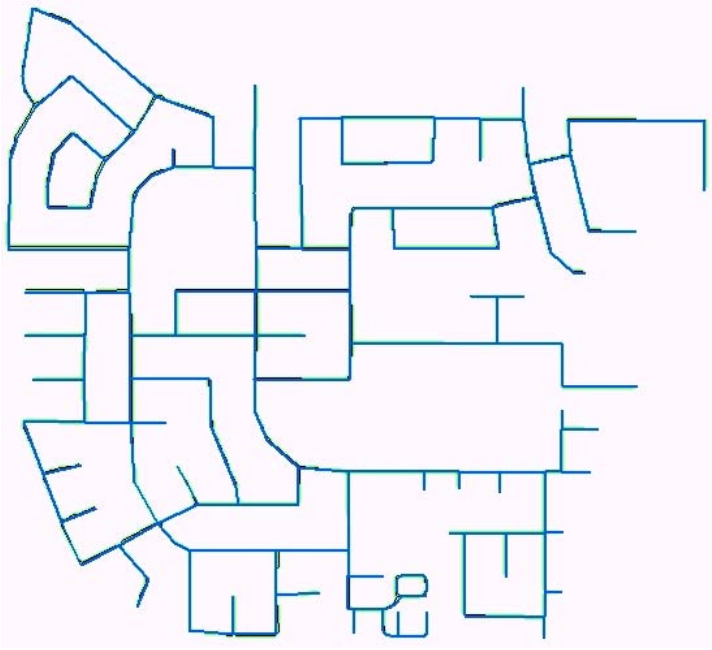

Figure 5.15: Overlay of Skeltonization by Mathematical Morphology on Scanned map 1

Then twenty well distributed matching points were selected and their respective coordinates were saved in a text file using ArcGIS. Using the original true coordinates and the coordinates collected from the two methods the mean and standard deviation

was calculated using the following formula:

$$\mu_{dx} = \sum_{i=1}^{n}(x_i - x_{true}) \tag{5.1}$$

$$\mu_{dy} = \sum_{i=1}^{n}(y_i - y_{true}) \tag{5.2}$$

$$\sigma_{dx} = \sqrt{\frac{\sum_{i=1}^{n}(x_i - x_{true})^2}{n}} \tag{5.3}$$

$$\sigma_{dy} = \sqrt{\frac{\sum_{i=1}^{n}(y_i - y_{true})^2}{n}} \tag{5.4}$$

$$\tag{5.5}$$

where μ_{dx} is the mean of the difference between the x coordinates and μ_{dy} is the mean of the difference between the y coordinates. similarly σ_{dx} and σ_{dx} are the standard deviation of x and y coordinates from the true values i.e., the estimated error or accuracy in the x and y direction respectively. In equaltion 5.3 and 5.4 the reason for using dividing by n and not n-1 is that we are using the true value of x.

The Root Mean Square error (RMSE)represents the difference between the original control points and the new control point locations calculated by the transformation process. The transformation scale indicates how much the map being digitised will be scaled to match the real-world coordinates [32]. In our case we will consider the well distributed 20 coordinates that we colled from the original map and the vectorized map using the three methods. In most of the cases RMSE is considered synonymus to standard deviation (in our case $RMSE_\sigma$). RMSE can be calculated by using the following formula:

$$RMSE_{\mu} = \sqrt{\mu_{dx}^2 + \mu_{dy}^2} \tag{5.6}$$

$$RMSE_{\sigma} = \sqrt{\sigma_{dx}^2 + \sigma_{dy}^2} \tag{5.7}$$

The mean and standard deviation of three methods for the first and second maps are shown in figure 5.16 and 5.17 respectively. In figures, 5.16 and 5.17 a) is Mean, Standard deviation and RMSE of Map 1 and manual digitization, b) is Mean, Standard deviation and RMSE of Map 1 and skeletonization by Mathematical Morphology and c) is Mean, Standard deviation and RMSE of Map 1 and skeltonization by Voronoi Diagrams.

From figure 5.16 and 5.17 we can infer that in our case manual digitization has resulted in the best accuracy, the second best accuracy is of skeletonization by Mathematical Morphology and the third best accuray is of skeletonization by Voronoi Diagrams.

5.3 Conclusion

Manual digitizing is tedious and time consuming. In this method the accuracy of maintaining the geometry of the object and digitizing in the center of the object completely depends on the operator. In this method everything from selecting the start node to end node and finding and correcting the dangling nodes is done by the operator. Skeletonization by Mathematical Morphology preserves the geometry of the object, forces the skeleton to be in the middle of the object and produces one pixel width skeleton and its accuracy is also upto 3.5 meters. The resulting image is then ready to be imported into ArcGIS or Grass to be converted to Vector data and

	x	*y*	RMSE
Standard deviation	0.3615	0.6203	0.7179
Mean	0.1059	0.1286	0.1665

(a)

	x	*y*	RMSE
Standard deviation	0.7559	1.1339	1.3628
Mean	0.2857	-0.4286	0.5151

(b)

	x	*y*	RMSE
Standard deviation	0.9047	1.3305	1.6089
Mean	0.4102	-0.6176	0.7414

(c)

Figure 5.16: Mean, Standard deviation and RMSE of Map 1

	x	*y*	RMSE
Standard deviation	0.1665	0.3759	0.4111
Mean	0.0989	0.1207	0.1560

(a)

	x	*y*	RMSE
Standard deviation	0.3780	0.7559	0.8452
Mean	0.1429	0.2857	0.3194

(b)

	x	*y*	RMSE
Standard deviation	0.5816	0.9851	1.1439
Mean	0.2055	0.4107	0.4592

(c)

Figure 5.17: Mean, Standard deviation and RMSE of Map 2

building the topology. Skeletonization by Voronoi Diagram preserves the geometry of the object, on an average the skeleton is approximately in the center of the object and the resulting data is in vector form and the topology is also built using the Quad edge data structure and its accuracy is also upto 5 meters. Therefore the main difference between both the skeletonizaion methods is that there is an extra step of importing and building the topology in Skeletonization by Mathematical Morphology, but, it results in the centerline of the object.

However it should be noted that after importing and building the topology in both the methods error checking has to be done manually. Hence, with respect to these the user can choose the vectorization method that best suits the needs of the GIS system he/she is using. For example, if time and accuracy is important then Skeltonization by Mathematical Morphology should be chosen as a first step towards vectorization. However, if an accuracy of upto 5 meters is desirable and time is important then Skeletonization by Voronoi Diagram should be chosen as a first step towards vectorization.

5.4 Summary

In this chapter comparison of both the methods are made based on the way it handles curves and junctions and its accuracy w.r.t the original data.

The next chapter gives the conclusion and gives further research directions. Conclusion gives a brief idea of what has been done in this research and future work gives a brief idea of how color maps and color satllite images can be vectorized.

Chapter 6

Conclusion and Future Work

6.1 Discussion and Conclusion

Paper maps have always been of prime importance in the field of surveying. In todays society there is a greater emphasis on digital data than on paper maps. This corresponds to the proliferation of GIS in industry and in the research environment which has lead to the need to convert all of the available analog geospatial data to digital form. Although the topological maps can be scanned, they cannot be used directly in a GIS system without processing. All the available commercial raster to vector conversion software are semi-automatic and are based on line tracing algorithms and therefore, require an operator. Thus automatic algorithms are required for faster and reliable conversion of maps.

A review of the existing commercial software for automatic vectorization has been done and their advantages and disadvantages are given in Chapter 2. All these commercial software require a perfect black and white scanned map to result in a good vector map. The quality of the output is directly proportional to the quality of the input. Editing raster data manually is much easier than editing vector data. Most of these software use raster editing tools that use mathematical morphological operations like pruning, dilation, erosion, etc. Almost all the software have the despeckle filter in their raster editing toolbox. Despeckle filter applies a soft, blurred effect to the image. It is especially effective on images with a high amount of contrast.

Thinning is chosen as the best method for vectorization of scanned map due to the reason that it extracts the centerline and preserves the shape and geometry of the object. Since thinning results in skeletons and skeletons are the centerlines of the objects that are thinned, we choose the two different methods for skeletonization, viz. skeletonization using Mathematical Morphology and Voronoi Diagram.

Thus the primary objectives of this research were to compare the two methods of thinning for extracting the centerlines of the lines (objects) on a black and white and grayscale scanned map, and filling gaps between the disconnected skeletonized lines using Least Square Parabola (LSP) fitting. The secondary objective was to suggest a methodology for linear feature extraction of satellite imageries using skeletonization and review the existing algorithms and the commercial softwares available for automatic vectorization of scanned maps.

Skeletonization by Mathematical Morphology has an accuracy of upto 3.5 meters and skeletonization by Voronoi Diagrams has an accuracy of upto 5 meters. Since skeletonization by Mathematical Morphology results in a raster image we have to import it in GIS software such as ArcGIS or GRASS and convert it to vector data and build the topology.

However it should be noted that after importing and building the topology in both the methods error checking has to be done manually. Hence, with respect to these the user can choose the vectorization method that best suits the needs of the GIS system he/she is using. For example, if time and accuracy is important then Skeltonization by Mathematical Morphology should be chosen as a first step towards vectorization. However, if an accuracy of upto 5 meters is desirable and time is important then Skeletonization by Voronoi Diagram should be chosen as a

first step towards vectorization.

Some of the applications that would benefit from this research are:

1. **Geospatial Information Systems:** Digitizing, Updating, etc.
2. **Remote Sensing:** Linear feature extraction in Satellite Imagery (roads, rivers, coastal boundaries etc.), Change detection, etc.
3. **Medicine:** Centerline extraction of the colon, arteries, intestine, etc.
4. **Robotics:** Steering algorithms (can help the robot to pass through the centerline and not collide with any obstacle), routing, etc.

The research contribution is shown in the next section.

6.2 Research Contribution

The contribution of this research are as follows:

- Comparison of skeletonization by Mathematical Morphology and Voronoi Diagrams
- Gap filling algorithm using Least Square Parabola (LSP) fitting
- Suggest a method for linear feature extraction using skeletonization from satellite imageries and suggest a pruning algorithm
- Analysing the existing commercial softwares available for automatic vectorization of scanned maps

The next section describes the future work.

6.3 Future work

6.3.1 Automatic vectorization of Color maps and Color satellite imageries

Color can be used in two ways. We can assume that the existence of three color components (red, green, blue) is an extension of the idea of a gray scale image or each color can be thought of as a special domain which consist of new information.

In automatic vectorization of color Images, we have to take into consideration that a color image is usually composed of Red, Blue and Green color composite. A color image is made of 3 layers which when seen individually appears to be as a gray scale image. So while vectorizing a color image one of the three bands that best highlights the feature of interested can be chosen for grayscale skeletonization. The algorithm suggested in section 3.4.1 can also be used for feature extraction and skeletonization.

Appendix A

Commercial Vectorization Software

1. Algolab - `http://www.algolab.com/`

   AlgoLab Raster to Vector Conversion Toolkit converts architectural, mechanical and various technical drawings, maps and other types of line artwork including black and white graphics for books and journals from raster to vector formats (CAD). The price of this software is 99 USD or 192 CDN. The toolkit incorporates a new advance Fine Line Technology that helps to create perfectly looking shapes and designs. It can refine curves. It is also tailored to digitize drawings including mathematical graphs for further use in MS Excel, Mathematica, Lotus 1-2-3 and other mathematical programs. Line artwork can be automatically recognized and represented in a vector format that then can be imported to CAD. The result of vectorization by AlgoLab Raster to Vector Conversion Toolkit is shown in figure 2.5.

   Supported Formats [1]

   Input Raster files of the following formats are supported:

   BMP, TIFF, JPEG, PNG, PCX, TGA.

   Output vector files can be saved in the following formats:

   DXF (AutoCAD), AI (Adobe Illustrator), EMF (Enhanced metafile) and WMF (Windows metafile) and TXT (ASCII XY) format importable into MS Excel

and other mathematical programs.

2. Able Software R2V - `http://www.ablesw.com/r2v/index.html`

   R2V for Windows is a raster to vector conversion software system. The system has an easy-to-use menu-driven graphical user interface in the Microsoft Windows environment. Because of raster and vector editing functions the software is well suited for applications in GIS, Mapping, CAD and scientific computing. R2V can be used for scanned maps and drawings, aerial photos and satellite imagery. The price of this software is 1,495.00 USD or 1,843.71 CDN. The result of vectorization by Able Software R2V for Windows is shown in figure 2.6.

   System Requirements [10]

   (a) Operating Systems: Windows 9x, NT, 2000 +

   (b) CPU: 486 or better

   (c) RAM: minimum 16 MB (more is recommended if large size images are processed)

   (d) HARD DISK: 4 MB for the software and extra space to store your scanned images.

   (e) DISPLAY: 8-bit SVGA capable of displaying 256 colors or better. If you are going to use the 3D display feature in R2V, your display should be set up using at least 16-bit or 24-bit color display.

   Supported formats [10]

   Input Raster files of the following formats are supported:

BMP, TIFF, JPEG, GIF, RLC, PNG, HDR.

Output vector files can be saved in the following formats:

GEN, PNT, MIF, DXF, SHP, IGS, SDL, XYZ

Raster to vector conversion software automated map digitizing and GIS data capture applications [10].

3. Arbor Image Corporation - `http://www.arborimage.com/aihome.htm`

   Draftsman series raster to vector software for standalone operation or as an add-on for AutoCAD [11].

4. AutoTrace - `http://autotrace.sourceforge.net/`

   A platform independent program for converting bitmap to vector graphics, distributed for free under the GPL [12].

5. celinea - `http://www.celinea.com/`

   CR2V converts raster images to vector graphics and exports into SVG and Postscript [8].

6. Consistent-Software - `http://www.csoft.com/`

   Hybrid graphics editing and raster to vector conversion software, as standalone programs and within AutoCAD.

7. Easy Trace - `http://www.easytrace.com/`

   Easy Trace PRO is a system for cartographic data management. It's a tool for quick editing of a big city's vector model. It's a tool that supports the entire

process flow sheet from scanning up to creating of the complete attributive coverage after multi-criterion verification. Plus object representation according to their attributive data values, plus means for work shear between many operators etc. [25]. The price of this software is 999 Euro (1,631.75 CAD). Unfortunately the demo software was not available so this software could not be tested.

It's a Cartographic raster-to-vector conversion software with tools for topology creation, checking and editing [25].

8. GTX Corporation - `http://www.gtx.com/`

   Tools for cleaning scanned documents for office and engineering purposes, as well as raster conversion software for AutoCAD and DXF [13].

9. International Origin Solutions - `http://www.origin-online.com/`

   Detector is a raster to vector converter that creates drawings and digitized maps out of raster images [51].

10. RasterTech - `http://www.rastertech.co.uk/`

    OEM supplier of software is for direct editing and raster-to-vector conversion of scanned images [54].

11. RasterVect-Software - `http://www.rastervect.com/`

    The raster to vector conversion program RasterVect allows you to transform scanned raster images into vector format on Windows systems [43].

12. Ravtek Inc. - `http://www.ravtek.com/`

Ravtek is the maker of Crucible, a powerfull easy to use automatic raster to vector conversion system [27].

13. ScanRaster - http://www.scanraster.com/

    It is a paper to CAD conversion software with an evaluation demo for download [46].

14. Sinclair Knight Merz - http://www.skmconsulting.com/spatial/cadproducts/

    ProVec Color is a raster-to-vector conversion product, with raster editing and transformation modules, compatible with common CAD systems.

15. SoftCover International Ltd. - http://www.softcover.com/

    Scan2CAD raster to vector conversion software with OCR text recognition and editing tools.

16. Softelec-GmbH - http://www.softelec.com/

    Providers of the HybridCAD software for raster to vector conversion and AutoCAD line tracing.

17. Softline-Software, Co. - http://www.softline-software.com/

    Scan2CAD PRO raster to vector conversion software with OCR text recognition and evaluation version for download.

18. SV ScanRaster - http://www.cadshop.co.uk/scanraster/

    Raster to vector conversion software for converting paper drawings into CAD.

19. Trix Systems Inc. - http://www.trixsystems.com/

TracTrix 2000 is a tool that enables you to edit and convert scanned drawings in AutoCAD. It uses the latest technology for automatic vectorizing, raster editing, rasterizing, calibration, text recognition etc. The price of this software is 750 USD or 925 CDN. The result of vectorization by TracTrix is shown in figure 2.4.

System requirements [28]

(a) A Pentium based system is recommended.

(b) Windows 95/98/NT 4/2000

(c) To install TracTrix 2000 as a plug-in into AutoCAD, one of the following platforms is required:

- AutoCAD LT 97, AutoCAD LT 98
- AutoCAD Map R2, AutoCAD Map R3, AutoCAD Map 2000
- AutoCAD R14, AutoCAD 2000
- AutoCAD Mechanical 14.5, AutoCAD Mechanical Desktop
- AutoCAD Architectural Desktop, AutoCAD Land Development Desktop
- A mouse or other Windows compatible pointing device
- VGA graphics or better
- A TWAIN compatible scanner (optional)

(d) Memory requirements:

- 10 megabyte (MB) free hard drive space for installation.

- 32 MB RAM minimum, 64 MB or more is strongly recommended
- To run text recognition a 200 MHz processor or higher is recommend

Supported Formats [28]

Opens the following raster file formats:

TIFF, BMP, CALS, JEDMICS C4, JPG and more.

Converts to the following vector formats:

DWG, DXF, IGES, AI(88), DRW (Micrografx file format), DXF (drawing interchange file), EPS (encapsulated postscript) AI-88, HPGL (Hewlett-Packard Graphics Language), IGES (Initial Graphics Exchange Specification) 5.1

Converts to the following raster formats:

BMP, CALS - Computer-Aided acquisition and Logistics Support, DIB - Device Independent Bitmap, PCX - Intel, EPS - Encapsulated Postscript, PNG, Tiff, WMF etc.

Products include scanner drivers, viewers, raster to vector and vector to raster conversion, and engineering document management [28].

20. VextraSoft - `http://www.vextrasoft.com/`

    Vextractor is a tool for transforming raster images into vector formats by building centerlines and outlines. This vectorizer tool could be used for vectorizing of charts, maps, schemes and other similar images for input to CAD and GIS systems. An application has an option to setup a reference for images, in other words to assign a real coordinates to certain raster points. Thus, you can get

vector data not only in the raster coordinates, but real geographical coordinates or in the chart coordinates [58]. The price of this software is 99 USD or 122.092 CDN. The result of vectorization by Vextrasoft is shown in figure 2.7.

Supported Formats [58]

Input raster formats:

BMP, GIF, TIFF, JPEG, PCX, TGA, PNG

Output format for vector information:

DXF, DXB, WMF, EMF, ArcView shapefiles, MapInfo MIF/MID, ASCII XYZ, Scalable Vector Graphic (SVG) Coordinate referencing

System Requirements [58]

Operating system: Win95/98/Me/NT/2000/XP TWAIN32-compatible scanners are supported

Vextractor is a raster to vector conversion tool for CAD and GIS systems [58].

21. WinTopo - `http://wintopo.com/softsoft/wintopo/index.htm`

    Raster vectorisation is based on both Zhang-Suen and Stentiford algorithm for skeltonization, as well as the Canny method for edge detection [31]. The price of this software is UK 200 (503.397 CAD). The result of vectorization by WinTopo Professional is shown in figure 2.8.

    Supported Formats [31]

    Input Raster Format

BMP - Bitmap files [*.bmp] TIFF - Tagged Image File Format [*.tif and *.tiff] and GeoTiff GIF - Compuserve GIF files [*.gif] JPEG - JPEG/JFIF Encoded pictures [*.jpg and *.jpeg] PNG - Portable Network Graphic files [*.png] WTX - WinTopo Project files [*.wtx]

Output Vector Format

DXF, SHP, MIF, Arc file format, DI GenaMap Type 19 Input file format, EMF Microsoft Enhanced MetaFile, WMF, ASC Ascii X, Y, Z format, WTX WinTopo Project Text file

A raster to vector conversion tool supporting several image and CAD/GIS file formats [31].

22. Geovect - `http://www.geovect.com/index.html`

    GeoVect specializes in the conversion of paper or raster maps into vector data for GIS, CAD or scientific analysis application. Their extensive knowledge of datum, spheroids, projections and data manipulation ensures the highest quality paper/raster to vector conversions available [20]. This software is a free ware. Unfortunately the demo software was not available so this software could not be tested.

    Geovect vectorize the following [20]:

    Engineering Drawings, Topographical Maps, Soil / Flood Maps, Parcel / Tax Maps, Lease Boundaries, Geographical Boundaries, Mechanical Drawings, Architectural Drawings, Plans and Profiles, Watershed Maps, Geological Outcrops, Benchmarks or Survey Points

They provide the following services [20]

(a) Free Scanning of Documents

(b) Free Shipping (US only)

(c) Competitive Rates

(d) Output to Media of Choice

(e) All Data Confidential

(f) Quick Turnaround

(g) 1 Year Backup of Data

(h) Free Online Storage of Data

(i) 100 percent Data Quality Guarantee

(j) 24/7 Online Support

(k) Corporate Accounts Welcome

(l) Volume Discounts Available

Bibliography

[1] Algolab. Algolab : Raster to vector conversion toolkit. `http://www.algolab.com/`, 2005. Accessed May 2005.

[2] John C. Antenucci, Kay Brown, Peter L. Croswell, Michael J. Kevany, and Hugh Archer. *Geographic Information Systems: A Guide to the Technology.* Van Nostrand Reinhold, 1991.

[3] F. Anton, D. Mioc, and A. Fournier. 2D image reconstruction using natural neighbour interpolation. In *The Visual Computer*, volume 17 No.3, pages 134–146, 2001.

[4] F. Anton, D. Mioc, and C. M. Gold. Line voronoi diagram based interpolation and application to digital terrain modelling. In *XXth ISPRS congress*, 2004.

[5] Patrice Arrighi and Pierre Soille. Sparse pixel vectorization: an algorithm and its performance evaluation. Proceedings of Geovision99, International Symposium on Imaging Applications in Geology, 1999.

[6] A. Baumgartner. Extraction of roads from arial imagery based on grouping and local context. In T. Schenk, A Habib, T. Schenk, and A Habib, editors, *Proceedings ISPRS Commission III Symposium, Vol. XXXII-3/1 of International Archives of Photogrammetry and Remote Sensing*, pages 196–201, 1998.

[7] J. L. Bentley and T. A. Ottmann. Algorithms for reporting and counting geometric intersections. volume C-28, pages 643–647, 1979.

[8] Celinea. Cr2v - celinea raster to vector converter. http://www.celinea.com/, 2003. Accessed May 2005.

[9] ESRI: Support center. Gis dictionary. http://support.esri.com, 2005. Accessed May 2005.

[10] Able Software Corp. R2v: Advanced raster to vector conversion software. http://www.ablesw.com/r2v/, 2005. Accessed May 2005.

[11] Arbor Image Corp. Arbor image: Leaders in raster to vector conversions. http://www.arborimage.com/aihome.htm, 2002. Accessed May 2005.

[12] Autotrace Corp. Autotrace-vectorization software. http://autotrace.sourceforge.net/, 2004. Accessed May 2005.

[13] GTX Corp. Gtx: Intelligent paper to cad solutions. http://www.gtx.com/, 2005. Accessed May 2005.

[14] C.Shih and R. Kasturi. Extraction of graphic primitives from images of paper based line drawings. volume 2 No.2 of *Machine Vision and Applications*, pages 103–113, 1989.

[15] D. Dori and W. Liu. Sparse pixel vectorization: an algorithm and its performance evaluation. volume 21 No.3 of *IEEE Transaction on Pattern Analysis and Machine Intelligence*, pages 202–215, 1999.

[16] C. M. Eastman. Vector versus raster: A functional comparision of drawings technologies. In *IEEE Computer Graphics and Applications*, pages 68–80, 1990.

[17] Bob Fisher, Simon Perkins, Ashley Walker, and Erik Wolfart. Hypermedia image processing reference.

[18] S. J. Fortune. A sweepline algorithm for voronoi diagrams. volume 2, pages 153–174, 1987.

[19] St. Frischknecht and E. Kanani. Automatic interpretation of scanned topographic maps: A raster-based approach. In Tombre K., Chhabra K. A., Tombre K., Chhabra K. A., Tombre K., Chhabra K. A., Tombre K., and Chhabra K. A., editors, *Graphics Recognition-Algorithms and Systems, Vol. 1389, Lecture Notes in Computer Science*, pages 207–220. Springer-Verlag, 1998.

[20] Geovect. Geovect: Raster to vector conversions. `http://www.geovect.com/index.html`, 2005. Accessed May 2005.

[21] C. M. Gold. Crust and anti-crust: A one-step boundary and skeleton extraction algorithm. In *Proc. 15th Annu. ACM Sympos. Comput. Geom.*, pages 189–196, 1999.

[22] C.M. Gold. The voronoi website. `http://voronoi.com/`, 2005. Accessed May 2005.

[23] Rafael C. Gonzalez, Richard E. Woods, and Steven L. Eddins. *Digital Image Processing using MATLAB*. Pearson Prentice hall, Pearson Education Inc., 2004.

[24] P. J. Green and R. R. Sibson. Computing dirichlet tessellations in the plane. volume 21 No.2, pages 168–173, 1978.

[25] Easy Trace Group. Easy trace: Advanced intelligent software for raster to vector conversion. http://www.easytrace.com/, 2003. Accessed May 2005.

[26] Leonidas J. Guibas and Jorge Stolfi. Primitives for the manipulations of general subdivisions and the computation of Voronoi diagrams. *ACM Trans. on Graphics*, 4 No.2:74–123, 1985.

[27] Ravtek Inc. Ravtek: Raster to vector conversion software. http://www.ravtek.com/, 2003. Accessed May 2005.

[28] Trix Systems Inc. Tractrix: Raster to vector converter, 2005.

[29] Gottfried Konecny. *Geoinformation: Remote Sensing, Photogrammetry and Geographic Information Systems*. Taylor and Francis Inc, 2003.

[30] George. B. Korte. *GIS Book*. Onworld Press, 1994.

[31] SoftSoft Ltd. Wintopo : Raster to vector conversion tool. http://wintopo.com/softsoft/wintopo/index.htm, 2005. Accessed May 2005.

[32] Susan Mayhew. *Dictonary of Geography*. Oxford University Press, 1997.

[33] D. Mioc, F. Anton, and G. Dharmaraj. An algorithm for centerline extraction using natural neighbour interpolation. In *XXth ISPRS congress*, 2004.

[34] Darka Mioc. The voronoi spatio-temporal data structure, 2002. University of Laval.

[35] R. B. Muhammad. Computaional geometry. http://www.personal.kent.edu/~rmuhamma/Compgeometry/compgeom.html, 2004. Accessed May 2005.

[36] J. R. Munkres. *Elements of Algebraic Topology*. Perseus Press, 1993.

[37] W. Keith Nicholson. *Elementry Linear Algebra with Applications*. Prindle, Weber & Schmidt, 1986.

[38] R. L. Ogniewicz. The skeleton space and its application to shape description and decomposition. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 746–751. WA, 1994.

[39] A. Okabe, B. Boots, and K. Sugihara. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. Wiley, 1992.

[40] N. Otsu. A threshold selection method from gray level histograms. *IEEE Trans. Systems, Man and Cybernetics*, 9:62–66, 1979.

[41] J. R. Parker. Gray level thresholding in badly illuminated images. In *IEEE transactions on pattern analysis and machine intelligence*, volume 12 No.8, pages 813–819, 1991.

[42] J. R. Parker. *Algorithms For Image Processing And Computer Vision*. Wiley Computer Publishing, 1997.

[43] RasterVect-Software. Rastervect-software: Professional graphics tools. `http://www.rastervect.com/`, 2005. Accessed May 2005.

[44] F. Rottensteiner. Object reconstruction in bundle block environment. In T. Schenk and A. Habib, editors, *Proceedings ISPRS Commission III Symposium, Vol. XXXII-3/1 of International Archives of Photogrammetry and Remote Sensing*, pages 177–183, 1998.

[45] S. Salvatore and P. Guitton. Contour line recognition from scanned topographic maps. *Journal of WSCG*, 12, 2004.

[46] ScanRaster. Scanraster - paper to cad conversion software. `http://www.scanraster.com/`, 2004. Accessed May 2005.

[47] J. Serra. *Image Analysis and Mathematical Morphology*, volume I & II. Academic Press, 1982.

[48] M. I. Shamos and D. Hoey. Closest-point problems. In *Proc. 16th IEEE Symp. Foundations of Comp. Science*, pages 151–162, 1975.

[49] S. S. Skiena. *The Algorithm Design Manual*. Springer Verlag, 1998.

[50] Consistent Software. Csoft: Consistent software. `http://www.csoft.com/`, 2005. Accessed May 2005.

[51] International Origin Solutions. International origin solutions. `http://www.origin-online.com/`, 2004. Accessed May 2005.

[52] Frischknecht St., E. Kanani, and A. Carosio. A raster-based approach for the automatic interpretation of topographic maps. In *Proceedings ISPRS Commission III Symposium, Vol. XXXII-3/1 of International Archives of Photogrammetry and Remote Sensing*, pages 523–529, 1998.

[53] John R Taylor. *An Introduction to Error Analysis: The Study of Uncertainties if Physical Measurements*. Maple-Vail Book Manufacturing Group, 1997.

[54] Raster Tech. Raster tech - giving you the right image. `http://www.rastertech.co.uk/`, 2002. Accessed May 2005.

[55] K. Tombre and Tabbone S. Vectorization in graphics recognition: to thin or not to thin. volume 2 of *Proceedings of 15th International Conference on Pattern Recognition*, pages 91–96, 2000.

[56] Montanari U. Continuous skeletons from digitized images. *Journal of the ACM*, 16:534549, 1969.

[57] Scott E. Umbaugh. *Computer Vision and Image Processing: a practical approach using CVIP tools*. Pearson Prentice hall, Pearson Education Inc., 1998.

[58] Vextrasoft. Vextractor: Raster to vector conversion tool. `http://www.vextrasoft.com/`, 2005. Accessed May 2005.

[59] Y. Wang and C. J. Trinder. Use of topology in automatic road extraction. In *Proceedings ISPRS, Commission III Symposium, Vol. XXXII-3/1 of International Archives of Photogrammetry and Remote Sensing*, pages 394–399, 1998.

[60] John. Weiss. Grayscale thinning. In *Proceedings of the 17th International Conference on Computers and Their Applications (CATA-2002)*, pages 86–89, 2002.

[61] John. Weiss. Grayscale thinning and ridge detection. In *Proceedings of the 15th International Conference on Computer Applications in Industry and Engineering (CAINE-2002)*, pages 107–110, 2002.

[62] E. W. Weisstein. Math world. `http://mathworld.wolfram.com`, 2004. Accessed May 2005.

[63] Liu Wenyin and Dov Dori. From raster to vectors: Extracting visual information from line drawings. volume 2 of *Pattern Analysis & Applications (1999)*, page

1021, 1991.

[64] C Wiedmann. Improvement of road crossing extraction and external evaluation of the extraction results. In *Photogrammetric Computer Vision, ISPRS Commission III, Symposium 2002*, 2002.

[65] C. Wiedmann and S. Hinz. Automatic extraction and evaluation of road networks from satellite imagery. In H. Ebner, W. Eckstein, C. Heipke, and H. Mayer, editors, *Proceedings ISPRS Conference on Automatic Extraction of GIS Objects from Digital Imagery, Vol. XXXII/3-2W5 of International Archives of Photogrammetry and Remote Sensing*, pages 95–100, 1999.

[66] S. Wu and A. Amin. Automatic thresholding of gray-level using multi-stage approach. In *IEEE, Proceedings of the Seventh Intenational Conference on Document Analysis and Recognition*, pages 493–497, 2003.

[67] Chunsun Zhang, Shunji Murai, and Emmanuel Baltsavias. Road network detection by mathematical morphology. In *Proc. of ISPRS Workshop "3D Geospatial Data Production"*, pages 185–200, 1999.