5-10-2003

Discrete Representation of Urban Areas through Simplification of Digital Elevation Data

Ruparani Chittineni

DISCRETE REPRESENTATION OF URBAN AREAS THROUGH SIMPLIFICATION OF DIGITAL ELEVATION DATA

By

Ruparani Chittineni

A Thesis
Submitted to the Faculty of
Mississippi State University
in Partial Fulfillment of the Requirements
for the Degree of Master of Science
in Computer Science
in the Department of Computer Science

Mississippi State, Mississippi

May 2003

DISCRETE REPRESENTATION OF URBAN AREAS THROUGH

SIMPLIFICATION OF DIGITAL ELEVATION DATA

By

Ruparani Chittineni

Approved:

Bharat K. Soni Adjunct Professor of Aerospace Engineering (Thesis Director)	Bradley D. Carter Professor of Computer Science (Major Professor)
Ioana Banisescu Associate Professor of Computer Science (Committee Member)	David Thompson Associate Research Professor, Engineering Research Center (Committee Member)
Julian E. Boggess Graduate Coordinator of Computer Science	A. Wayne Bennett Dean of the College of Engineering

Name: Ruparani Chittineni

Date of Degree: May 10, 2003

Institution: Mississippi State University

Major Field: Computer Science

Major Professor: Dr. Bradley D. Carter

Title of Study: DISCRETE REPRESENTATION OF URBAN AREAS THROUGH SIMPLIFICATION OF DIGITAL ELEVATION DATA

Pages in Study: 63

Candidate for Degree of Master of Science

In recent years there has been large increase in the amount of digital mapping data of landscapes and urban environments available through satellite imaging. This digital information can be used to develop wind flow simulators over large cities or regions for various purposes such as pollutant transport control, weather forecasts, cartography and other topographical analysis. It can also be used by architects for city planning or by game programmers for virtual reality and similar applications. But this data is massive and contains a lot of redundant information such as trees, cars, bushes, etc. For many applications, it is beneficial to reduce these huge amounts of data through elimination of unwanted information and provide a good approximate model of the original dataset. The resultant dataset can then be utilized to generate surface grids suitable for CFD purposes or can be used directly for real-time rendering or other graphics applications. Digital Elevation Model, DEM, is the most basic data type in which this digital data

is available. It consists of a sampled array of elevations for ground positions that are regularly spaced in a Cartesian coordinate system. The purpose of this research is to construct and test a simple and economical prototype which caters to image processing and data reduction of DEM images through noise elimination and compact representations of complex objects in the dataset. The model is aimed at providing a synergy between resultant image quality and size through the generation of various levels of detail. An alternate approach using the concepts of standard deviation helps in achieving the desired goal and the results obtained by testing the model on Salt Lake City dataset verify the claims. Thus, this thesis is aimed at DEM image processing to provide a simple and compact representation of complex objects encountered in large scale urban environment datasets and reducing the size of the dataset to accommodate efficient storage, computation, fast transmission across networks and interactive visualization.

DEDICATION

To my family and friends.

ACKNOWLEDGMENTS

I would like to thank Dr. Bharat Soni for assisting and guiding me throughout my masters' program. I would like to thank Dr. Brad Carter for providing valuable support. I would like to thank my committee members Dr. Ioana Banisescu and Dr. David Thompson.

Most of the ideas in this thesis is a result of the numerous discussions I have had with Niranjan Sharma. Also, Eric Collins has been very forthcoming in guiding me on the design issues of this work. I would like to thank them for their invaluable suggestions and help.

Thank you,

Rupa

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

One of the long-standing aims of science has been to be able to understand and predict 'nature'. By developing computer simulations of nature, combined with computational simulation of the forces that act upon it, science is aiming to provide mankind with the power to predict and hence prevent hazardous events. One such tool is the Wind Field Simulator.

Wind Field Simulator is a tool that can help understand and predict chemical/pollutant transport through wind in the environment. In recent years, the inclination towards building pollutant-dispersion simulators for urban areas is on the rise because of its high usability and need. For example, in case of toxic warfare, evacuation plans can be developed using dependable simulation tools. Pollution monitoring and environment management are other highly useful capabilities of wind field simulations. Resource management through control of forest fires is also possible using a Wind Field Simulator. Other applications such as cartography and city planning would also benefit from computer simulations in a way that helps in understanding the spatial characteristics and hence assist in rural infrastructure design and urban development.

1.1 Overview of the Wind Field Simulator

The Wind Field Simulator is essentially a software program, which numerically recreates the gross features of the wind [6]. The simulation thus requires the interaction of multiple disciplines: wind data from various meteological stations available throughout the region, wind prediction (obtained by numerically solving mathematical models based upon finite-element, finite-volume and boundary-element methods using wind data and assimilating models with experimental data), numerical simulation of mathematical models representative of chemical pollutants in question and influence of structures (eg. buildings), plants and hazardous sites in the area (see figure 1.1).

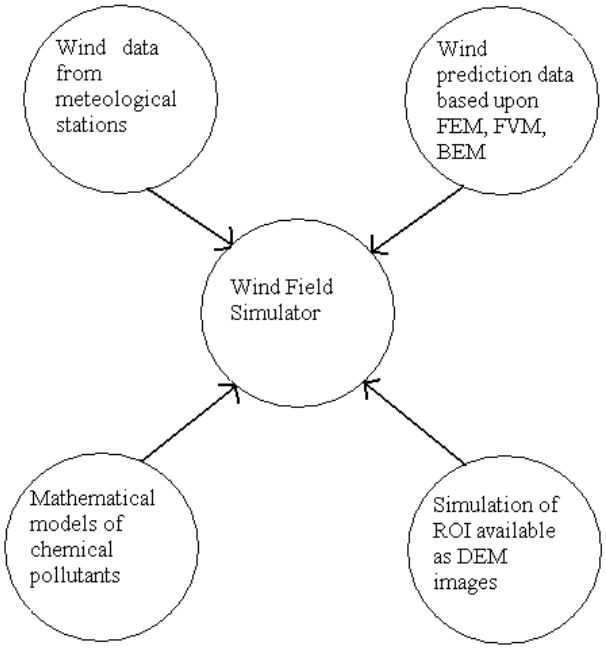

Figure 1.1 Multidisciplinary Input to Wind Field Simulator

With the advent of high performance computing and communication and robust and efficient mathematical/numerical algorithms, this simulation has been made possible. A multitude of techniques and software, which employ Gaussian plume, Lagrangian or Eulerian models, have been developed to perform disciplinary computational simulations on the given region of interest (ROI). However, due to the enormous size of the input datasets to these simulators, the response times are not very satisfactory. The large size for description of the ROI can be attributed to the reduntant representation of the objects (for example, if a cube representing a simple building can be represented using four points of certain elevastion then in the DEM image it is being represented by many more points). Also, the presence of irrelevant entities which would not contribute to the accuracy of the simulations adds to the execution times. Such entities are referred to as 'noise'. Thus, noise could be defined as objects/entities observed in the representations of the regions of interest that are of least significance to the simulation process. For example, contribution by objects such as cars, trees, bushes, towers or lamp-posts to the simulation process is negligible and hence can be safely eliminated. However, presence of the same can introduce huge errors during approximation while also increasing the size of resultant dataset.

Thus, a need for generating good approximations to an existing definition of ROI, that is input to the simulator model is realized.

The information about the region may be captured as satellites images and converted to a form appropriate for numerical processing[2, 5, 11]. After conversion, the

description may be stored in the form of a set of points, a set contour vectors or a set of irregularly spaced set of points connected as triangles also called Triangular Irregular Network (TIN) Model as shown in figure 1.2. The most commonly available form of description is the point form. Each point represents its elevation from the ground level. Each of these representations has their advantages and disadvantages. But they share a common drawback : the data describing the concerned region is very huge and contains noise. The next section describes the digital elevation model.

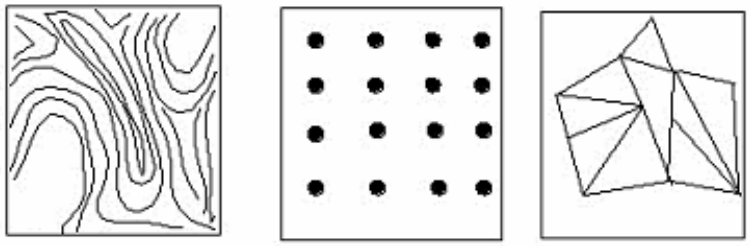

Figure 1.2 Forms of DEM Representation: Contours, Points, Triangles

1.2 Overview of the Digital Elevation Model

The Digital Elevation Model (or DEM) consists of a regular grid of spot heights, i.e., it consists of elevation data over a horizontal grid of one meter or more resolution [2, 9]. This data may be derived from ground survey, photogrammetry or cartographic data capture. The Digital Elevation Model is calculated using parallax measurements/stereoscopy [2]. These digital cartographic/geographic data files are sold in 7.5-minute, 15-minute,

2-arc-second (also known as 30-minute), and 1-degree units. The 7.5- and 15-minute DEMs are included in the large-scale category while 2-arc-second DEMs fall within the intermediate scale category and 1-degree DEMs fall within the small-scale category. This data is then polished or preprocessed to remove any discrepancies or errors such as overlapping buildings, gaps and unfinished sidelines induced due to the processing of satellite images through stereoscopy.

But this preprocessing does not reduce the presence of unwanted objects such as trees, bushes, lamp-posts. These datasets can be very huge (e.g., the data file of New Orleans consists of 12 million points while a 4400 X 5700 sq m section of Salt Lake City consists of 20 million points). A snapshot of a part of the Salt Lake City dataset is shown in figure 1.3. Such huge datasets would inadvertently prove to be very time consuming and CPU intensive for the model solvers.

In view of risk mitigation (i.e., disaster-control), related simulations such as in toxic warfare or control of forest fires, where time is the deciding factor for the extent of damage, small datasets with good approximation models of the region of interest prove to be highly useful. Thus, to address the issues of risk mitigation,(i.e., timely response towards disastrous situations), it is essential to reduce simulation and rendering times. There can be two approaches to the problem:

1. the rendering engine should be manipulated/modified to accommodate the needs of the simulator, or
2. the size of the datasets being sent to the rendering engine must be reduced.

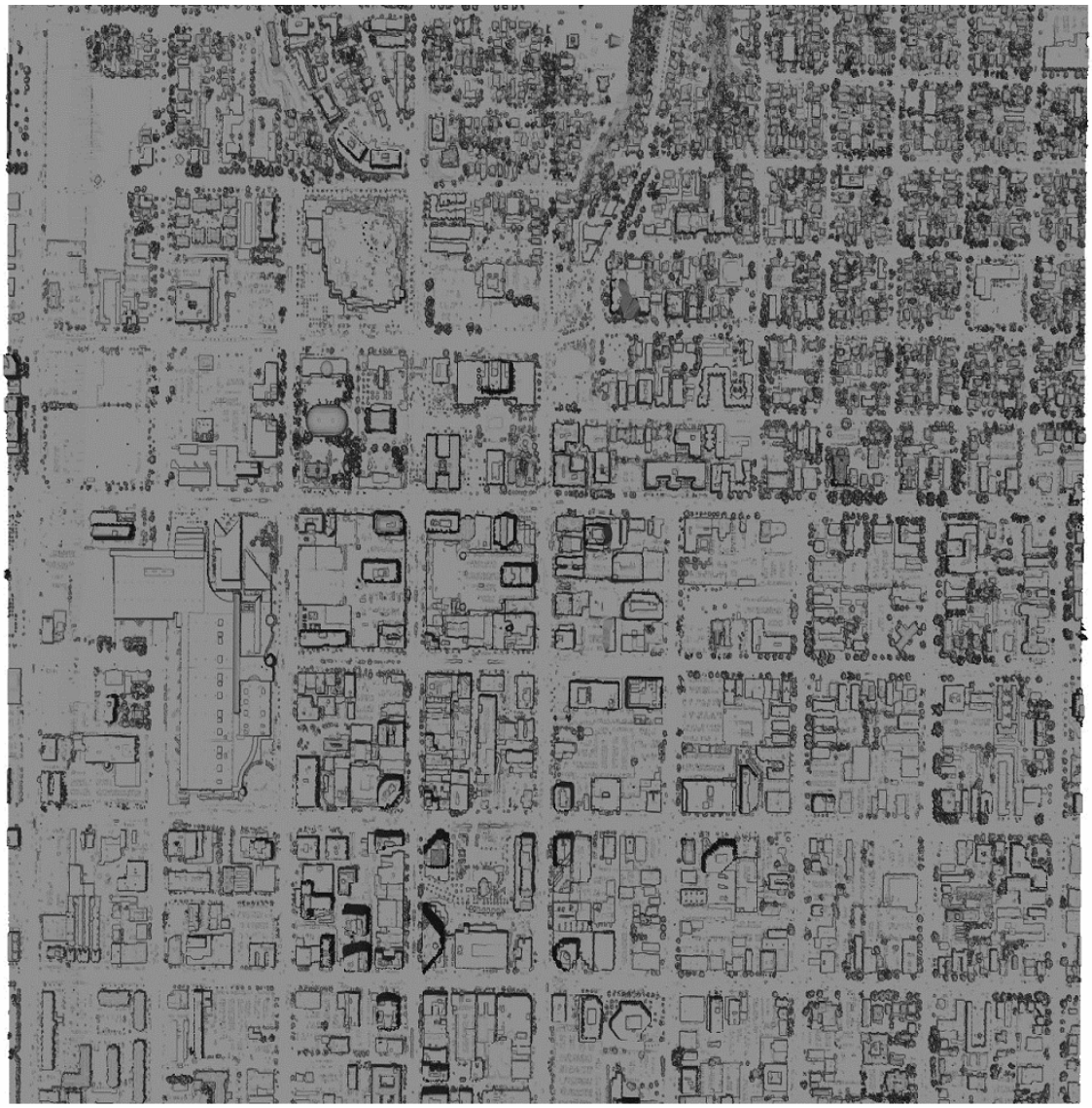

Figure 1.3 Part of the Salt Lake City dataset consisting of over 20 million points

This research will explore the second approach where discrete representation and reduction of the DEM dataset with experimentation of the application of a modified scan-line and marching cubes algorithm.

This research is an attempt at eliminating noise from the given dataset for an urban environment simulation and using simple approximate representations for complex objects/structures. Thus by reducing the size of the dataset provided to the Wind Field Simulator, the response times could be enhanced and interactive visualization of the pollutant transport could also be feasible.

1.3 Research Issues

There are numerous algorithms available in the literature that cater to image processing and more specifically to DEM data reduction. The Triangular Irregular Network (TIN) Model [15] and Contouring [17] are the most commonly used techniques. The TIN Model consists of irregularly spaced sample of points connected by triangles and only those points are included which largely contribute to shape enhancement of the structures and do not introduce noise. But as will be learnt in the proceeding chapter, this technique is more suitable for terrain datasets or landscapes. The characteristics of terrain datasets is that they do not contain structures with sharp edges or extremely steep slopes as in case of urban areas with buildings and other man-made structures. Mountainous regions, plateaus or hilly regions are more suited to polygonal simplification using the TIN Model representation.

Volumetric Rendering using Marching Cubes [17] and Contouring are other alternatives which would assist in reducing the size of the input dataset, but they would not be able to capture the edge information of buildings accurately and also the triangulations produced for rendering may contain triangles which are smaller than the pixel size itself. Thus numerous unwanted triangles would be generated. Some of the other techniques such as the Uniform Grid Algorithm, Heirarchical Methods and the Feature Methods Algorithm also cater to DEM processing and will be discussed in greater detail in the next chapter.

1.4 Hypothesis/Research Goals

It is clear that a new methodology, which would cater to urban environment simulations, would be extremely valuable. Thus a modified scan-line algorithm, similar to the Marching Cubes Algorithm, is being proposed in this research to customize the problem of DEM simplification for urban datasets. Specifically, the goal of this research is to develop and test a modified scan-line algorithm that will

1. eliminate noise (automobiles, trees, bushes, etc.), and
2. provide an accurate approximate representation of complex objects in the dataset at various levels of detail.

1.5 Approach

The approach in this research is to outline the essential elements that would form the outline of the new modified data reduction algorithm. The main elements of the algorithm are

1. 'cleaning' the geometry by eliminating the unwanted objects such as cars, trees, bushes, etc. from the dataset,
2. segregating the buildings so that an object oriented data structure which would cater to component modeling could be achieved, and
3. providing support for the generation of various levels of detail for the given dataset.

To obtain the exact shape of the building or its approximation, a point distribution curve, based on 'standard deviation' calculation, will be used to determine the spatial distribution of the cross-sectional planes. The convex hull of the building at each of these cross-sections will be captured and stored. The various levels of detail combined with a noise-control function will facilitate the generation of good approximation for urban environment datasets.

The algorithm will then be evaluated for its accuracy by comparing certain statistics of the original structures or objects in the dataset to those obtained using the data reduction algorithm. Also, a visual comparison of the raster image of original dataset with the resultant approximations of the same at various levels of details will be used as a measure of evaluation for the proposed algorithm.

For experimentation purposes, a part of the Salt Lake City dataset will be taken as a test case and will be a running example for the entire report. The dataset consists of

elevation information for the entire Salt Lake City at one meter resolution. For simplicity in storage and use, the entire region will be divided into five sections. Each section will contain approximately 20 million points of which small sections of 242 x 249 and 70 x 85 points will be considered as test cases.

The proposed algorithm could be applied to terrain datasets as well, but certain characteristic features of the structures that would be present in an urban setting are taken into consideration to provide a simpler solution to the problem and thus customize it to a certain extent.

1.6 Organization of Thesis Report

The organization of the report is as follows: Chapter 2 outlines the numerous methodologies available in literature that cater to DEM image processing. It also provides a tabular representation of the advantages and limitations of each technique. Chapter 3 summarizes the attempts at DEM approximation using existing in-house and free software tools. Chapter 4 provides the framework for an alternate approach to simplification and provides a step-by-step explanation of the algorithm developed. Chapter 5 describes the results obtained using the prototype when applied to Salt Lake City dataset and also states the complexity of the algorithm. Finally, Chapter 6 summarizes the characteristics of the alternate model and outlines the scope for future work.

CHAPTER II

LITERATURE REVIEW

Image processing has become increasingly important since the past decade as satellite photographs and scanners for remote sensing, scientific visualizations and other similar applications are capturing datasets of greater and greater detail. Analyzing these datasets in their original form is frequently painstaking, cumbersome, time consuming and CPU intensive.

Simplification, while retaining characteristic features, is very beneficial in terms of storage, transmission and computation. In other words, a compact approximation of the original dataset will reduce disk and memory requirements and also improve network related operations. Numerical simulations, such as in Finite Element Analysis, with execution times directly proportional to size of input datasets are accelerated. Also, real-time rendering becomes possible in the truest sense and, hence, simplified datasets are appropriate for risk-mitigation applications.

The literature offers numerous methodologies, that cater to image processing of DEM data through polygonal simplification and approximation. These methodologies have been applied in varied fields such as computational geometry, computer graphics, geographical information systems (GIS), virtual reality, finite element methods and car-

tography. A detailed description of the various techniques for polygonal simplification is available in a survey paper by Garland and Heckbert [15].

This chapter provides a review of key techniques for simplification that revolve around DEM data reduction and approximation of urban datasets. The following five algorithms are discussed:

1. regular subsampling algorithms
2. hierarchical subdivision algorithms
3. feature methods algorithms
4. refinement and decimation algorithms
5. marching cubes and contouring algorithms

2.1 Regular Subsampling Algorithms

Regular subsampling is the simplest and fastest approximation algorithm that can be implemented [14]. Image processing in this case involves generating the desired the approximation through scanning the entire dataset, retaining every point in the kth row and kth column and discarding the remaining points [14]. As shown in the figure 2.1, various levels of details (LOD) are achieved by simply subsampling the original dataset at regular intervals in the horizontal and vertical directions. The resultant datasets are also called uniform grids. The DEM dataset is also also referred to as a regular grid and the next lower level of detail is obtained by subsampling to a quarter of the number of points in the parent LOD.

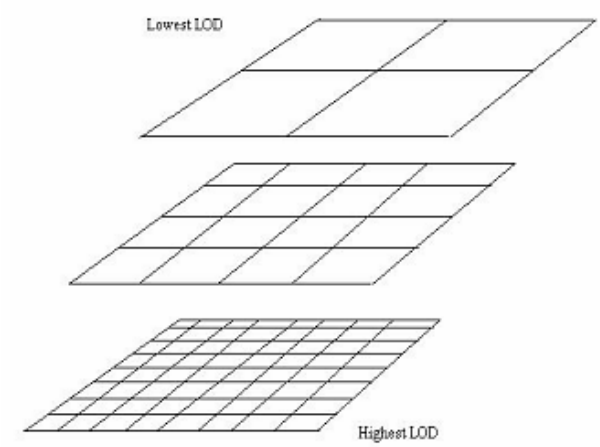

Figure 2.1 Levels of detail obtained using Regular Subsampling

As is obvious from the description, regular subsampling may lose vital information in the process of generating LODs. The discarded points could be the most important and could be the 'characteristic' point that defines the particular shape of the object. Hence, regular subsampling does not prove to be a good choice for approximation.

2.2 Hierarchical Subdivision Algorithms

Hierarchical subdivision is another relatively simple and fast technique. It consists of hierarchically subdividing a surface while generating a triangulation of the surface [1, 14, 15]. The triangulation can be quaternary as shown in figure 2.2 or ternary as shown in figure 2.3

The mid-point of each edge is connected to obtain the next level of detail. This technique is adaptive in the sense that it generates various LODs in a perspective manner,

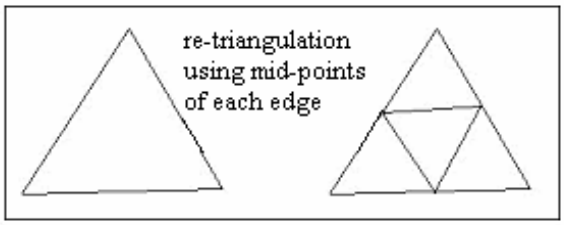

Figure 2.2 Quaternary Triangulations

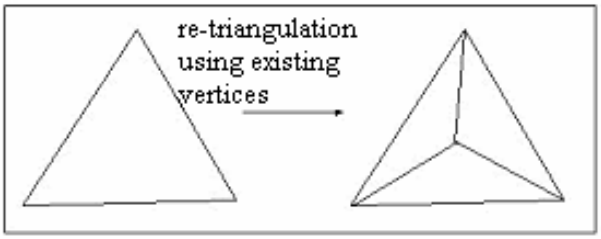

Figure 2.3 Ternary Triangulations

i.e., nearer terrains or objects can be displayed in greater detail than the farther ones through controlled subdivision. Quad-trees and k-d trees are the most common forms of representation for hierarchically subdivided datasets. The nodes or leaves of the tree represent the cells of the datasets (figure 2.4).

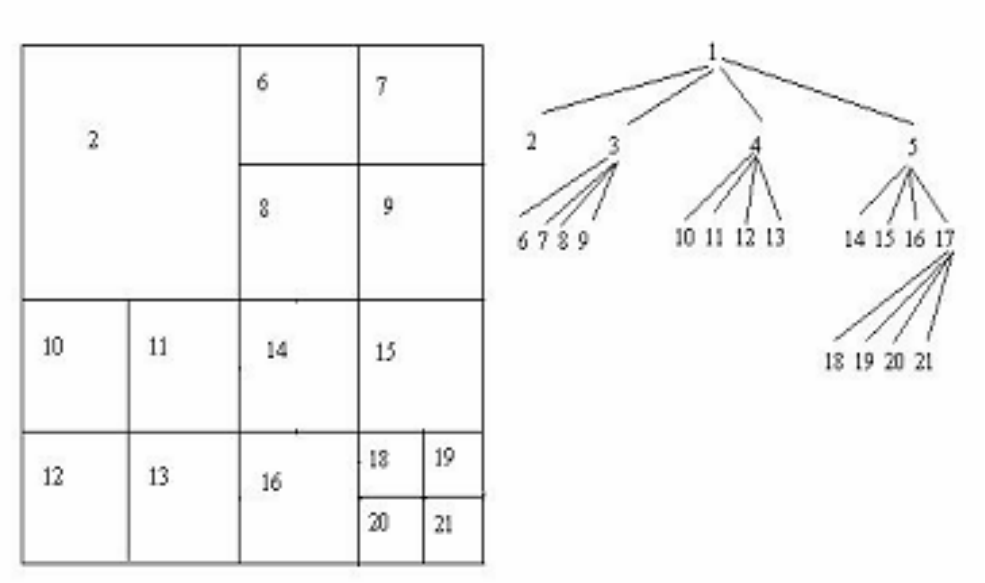

Figure 2.4 Hierarchical Subdivision: Quad tree data structure

But the penalty for this simplicity is the quality of approximation. It typically yields poor approximation. A slight modification to the traditional approach would involve computing the point of highest error along each triangle edge and in the interior of the triangle. The points with error above the threshold would be the new vertices.

The cost of this algorithm was estimated to be in the order of O(n log m) but if the hierarchy is very unbalanced then the worst case cost is O(mn). In the case of

approximation of an urban environment, the edge information of buildings may not be captured well.

2.3 Feature Method Algorithms

Another simple and intuitive algorithm which can be used for image processing is the one-pass or multiple-pass feature method [15]. In this technique, a list of the most 'important' or 'feature' points (based on some metric such as height) is generated over a single pass or multiple passes. This list becomes the vertex list that is used for triangulating the entire surface. One such method is the Triangular Irregular Network (TIN) Model.

The Triangular Irregular Network (or TIN as it is most popularly known) consists of irregularly spaced sample of points connected to form triangles (see figure 2.5). There could be more points, hence more triangles, to represent the smooth surface while fewer number of points to represent a coarse terrain or surface. Using feature methods, various levels of detail of TIN can be generated by iteratively adding vertices from the vertex list and retriangulating using Delaunay triangulation or data-dependent triangulation [10]. The most commonly used feature detectors are the 2 x 2 or 3 x 3 linear or non-linear filters. Feature methods are mostly used in cartography. As in the previous case, this technique cannot be applied to our scenario of urban area simulation because of the presence of sharp-edged structures as opposed to arbitrary forms of mountainous, hilly

or forested areas. Also the use of 2 x 2 or 3 x 3 filters makes the algorithm more prone to noise and other high frequency variations [14].

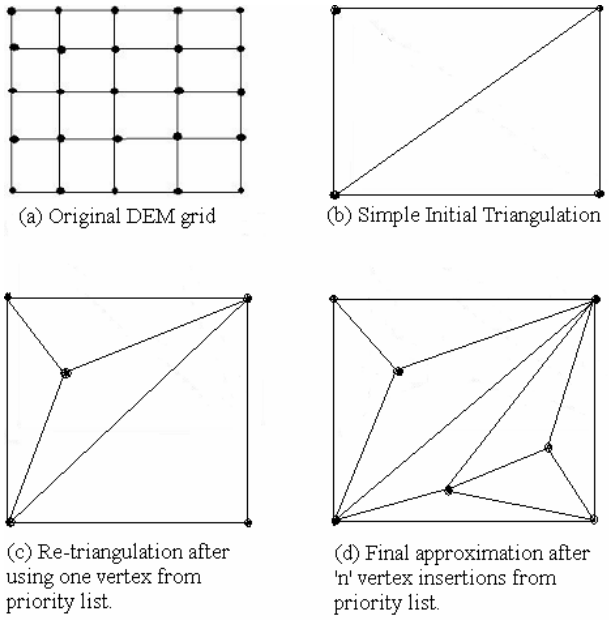

Figure 2.5 Triangular Irregular Network Model.

2.4 Refinement and Decimation Algorithms

Refinement methods revolve around iteratively introducing vertices of highest priority to generate the next LOD of TIN. The algorithm starts with an initial coarse triangulation of the entire domain (dataset). Then by either using a 2 x 2 or 3 x 3 filter, as used in the feature method algorithm, a candidate list or vertex list is generated [12, 15]. For this, the error input at each point is computed and tested against the highest error

seen so far for the triangle. Single points or multiple points may be added throughout the triangulation (see figure 2.6). On each pass, the set of triangles is scanned for the candidate of highest error. Finally, using Delaunay or data dependent triangulation, the entire surface is obtained. This method is similar to the previous one except for the fact that the importance of a point is usually a measure of error between it and the approximation. Feature methods are entirely independent of the approximation.

Figure 2.6 Refinement Methods Through Point Insertion

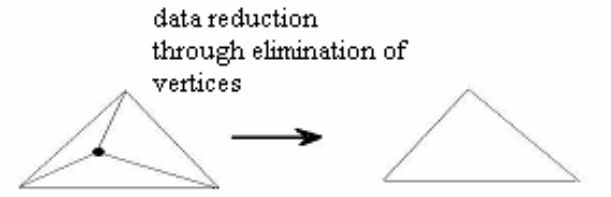

Figure 2.7 Decimation Method through Edge Collapse

Decimation methods [18], on the other hand, start off with the finest triangulation of the entire dataset and iteratively delete vertices using edge collapse or vertex split

to obtain the best approximation after simplification. Figure 2.7 illustrates the results of the algorithm. The cost of this algorithm is relatively high compared to refinement methods since the procedure involves the entire datasets from the first stage. Storing the connectivity of all points of the entire dataset consumes a lot of memory.

Refinement and decimation methods can perform data reduction to a satisfactory level. Many variations of these techniques have been cited in literature [15] and also have been successfully implemented in various GIS and cartographic applications. The technique has also been employed for processing DEM images of the Salt Lake City dataset and will be described in next chapter. The results obtained from this experimentation triggered the need to investigate another approach to the problem of DEM processing that would cater specifically to an urban setting i.e. with specific structural features, and hence generate better approximations while keeping the noise level to a low.

2.5 Marching Cubes and Contouring Algorithms

Marching cubes and contouring algorithms generate contours around the region of interest based on user -defined threshold levels [17]. This means that the 'importance' of a point is user-defined. Thus it would be beneficial if the user had some knowledge of the region of interest. By selecting an appropriate threshold level, the noise is filtered from the relevant regions in the image and a contour is drawn around it. The points below the threshold are discarded while those above are retained to be connected to form contours

or triangles. The pixel information at each voxel corner is read and determined to be inside or outside the surface. For edges with one corner outside the surface and one corner inside the surface, the intersection point is retained. Certain ambiguities have been observed in the triangulation of these points and resolving them would increase the complexity of the algorithm. At the same time, the triangulations may result in certain triangles being smaller than the pixel size itself. Also computing and retaining intersection points would amount to increase in the size of the dataset

2.6 Summary

Table 2.1 gives a summary the existing data reduction algorithms that apply to DEM image processing.

While all the techniques described above have their advantages and limitations, none of them can accurately capture the straight edges of buildings or distinguish clearly between noise and region of interest. A prototype model that aims to address these issues and provide a synergy between resolution of approximations and faster processing, i.e., smaller dataset, is required.

Table 2.1 A summary of existing data reduction algorithms

Method	**Advantage**	**Limitation (w.r.t.DEM image processing)**
Regular Subsampling	simple and economical	points discarded during subsampling could be the most important ones
Hierarchical Subdivision	can generate multi-resolution surfaces. More triangles for representing smooth surface and fewer triangles for coarse region.	building edges may not be accurately captured.
Feature Methods	slopes are efficiently captured because of the use of 3 x 3filters	city datasets are characterized by sharp edges that are difficult to capture or distinguish from noise using 3 x 3 filters.
Refinement and Decimation	good quality approximations when applied to terrains	extent of noise elimination is not good and steep-edge capturing capability is not satisfactory
Marching Cubes and Contouring	captures the exact shape for given threshhold	extra vertices/points are introduced as a result of computing intersection points and this adds to the size of the dataset, considerably.

CHAPTER III

INITIAL EXPERIMENTATION

For a proposal entitled, ”Wind Field Simulation Over Salt Lake City” submitted to Risk Management Program (RMP) [16], numerous experiments were performed using existing software packages to generate surface grids suitable for performing computational simulation over the Salt Lake City dataset [16]. The Salt Lake City dataset consisted of DEM data of 1m-resolution or elevation information over the entire city. The data was divided uniformly and stored over six CDs, each containing approximately 6500 x 5000 sq. m. of area and hence having approximately over 20-30 million points.

To further simply the size of the dataset for experimentation, a small part of the city with a grid size of 2019 x 1801 having 3.6 million points was considered (see figure 1.3). The problem definition was to process the DEM image and capture the buildings in the Salt Lake City dataset while keeping noise at a minimum. Two approaches to the problem were considered for experimentation:

1. mesh simplification
2. surface fitting using NURBS

Each of the techniques is discussed in detail in this chapter.

3.1 Mesh Simplification

The approach discussed here use mesh refinement and mesh decimation techniques [10, 11, 12, 16] defined in the earlier chapter to generate TIN models of the region of interest.

3.1.1 *Mesh Refinement*

TERRA, a software package developed at Carnegie Mellon University, utilizes concepts of mesh refinement [13]. An initial triangulation of the DEM dataset with few of the boundary points is obtained. Using points from the priority list, the next levels of detail are created. The iterations end when a certain limit on the number of points that may be present or the certain error tolerance is reached.

To generate the priority list, the researchers at the Engineering Research Center (ERC) at Mississippi State University used gradient values of the points [14]. The gradients were determined using Sobel Operators.

The Sobel Operator [3] performs a 2-D spatial gradient measurement on an image and so emphasizes regions of high spatial frequency that correspond to edges. The operator consists of a pair of 3 x 3 convolution kernels as shown in figure 3.1.

The kernel on the right is simply a 90-degree rotated version of the one on the left. These kernels are designed specifically to respond maximally to edges running vertically and horizontally relative to a particular point, one kernel for each of the two perpendicular directions. The two components can then be combined to obtain the absolute gradient and its orientation at each point in the grid. The magnitude is given by

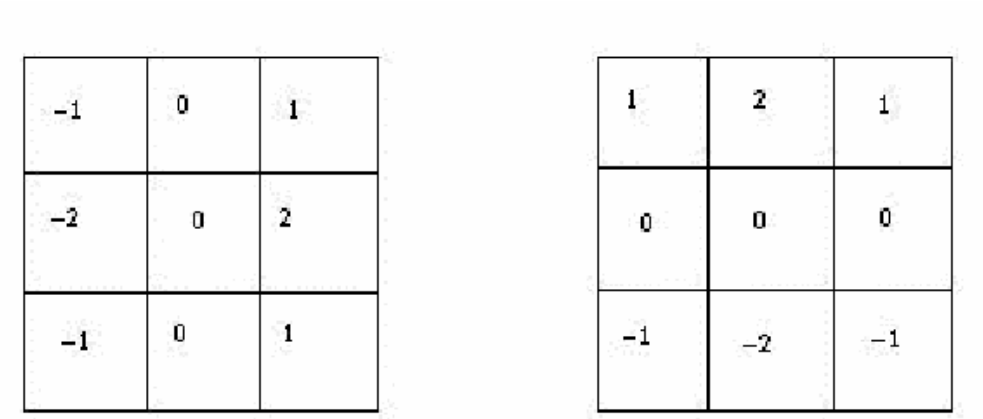

Figure 3.1 Sobel Operators

modulus of H = sqrt (hx2 + hy2)

and the direction is given by

q = arctan(hy/hx).

An orientation of zero implies that the direction of maximum contrast from minimum to maximum height runs from left to right on the grid. Other angles are measured counter-clockwise from this. Sobel operators are less sensitive to isolated, high intensity point variations since local averaging over sets of three points/pixels tends to reduce this. In effect, this is a 'small bar' detector as opposed to point detector as can be seen from figure 3.2. Thus a lot of noise can be eliminated and using the magnitude of gradient, the points could be assigned an order of importance or priority.

Having thus generated the priority list, the LODs of the TIN Model is obtained. The results of this procedure when visualized using the TERRA package are shown in figures

Figure 3.2 Edge Detection using Sodel Operators

3.3 and 3.4. As can be seen, the sharp edges of the buildings have not been captured well. Also, the noise levels are fairly high.

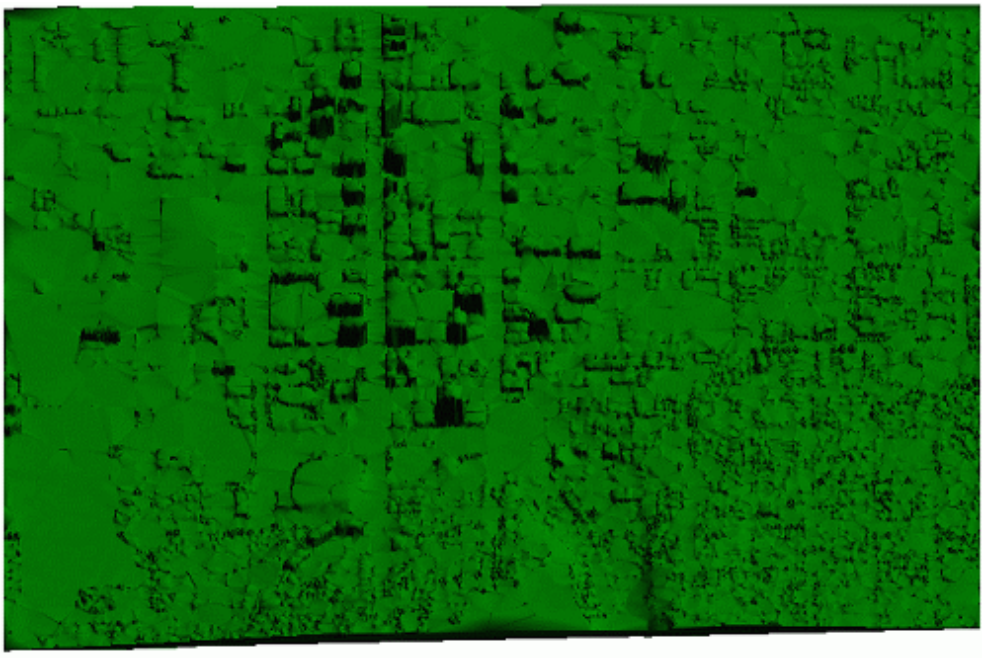

Figure 3.3 Mesh Refinement : Visualization of TIN using 'TERRA' Package

3.1.2 *Mesh Decimation*

Decimation techniques involved triangulating the entire dataset, initially and the iteratively deleting vertices of least importance (see section 2.4). The commercial software package, VTK [19] was used for this purpose. The tool could not upload the entire region of interest, so a patch of 242 x 249 constituting the Main Square region of Salt Lake City dataset was considered. The results are shown in figures 3.5 and 3.6. Neither

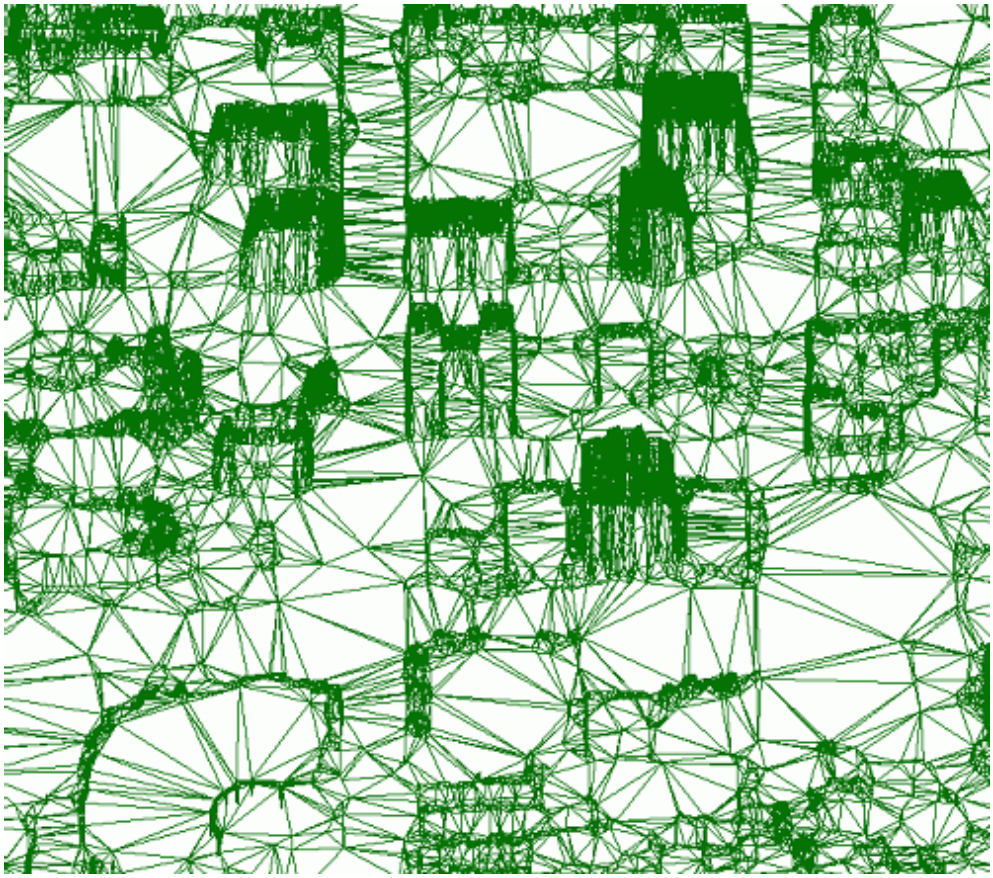

Figure 3.4 Mesh Refinement: Wireframe Model of Figure 3.4

could the number of points be reduced considerable (further decimation could not be done without compromising on topology) nor the edge information could be captured accurately.

3.2 Surface Fitting

By using the grid points as control points of non-uniform rational b-splines (NURBS), a surface grid of the region of interest can be obtained. In this regard, SOLIDMESH, grid generation software developed at the ERC was used. A subset of the region of interest, constituting of approximately 242 x 249 points was considered and the Nurbs surface was generated.. The results obtained after performing grid generation over the NURBS surface is as shown in the figure 3.7 and 3.8.

Though the approximation of the images generated was good, problems were encountered at sharp edges. Surfaces were also noted to be overshooting.

3.3 Summary

Table 3.1 provides a summary of the initial experimentations discussed above.

Thus, while surface-fitting techniques provided good quality approximations, they lack the ability to provide considerable data reduction. On the other hand, mesh simplification techniques reduce the original dataset to give various levels of detail with each level having considerably small number of points, but the sharp edges of the building structures were not accurately captured.

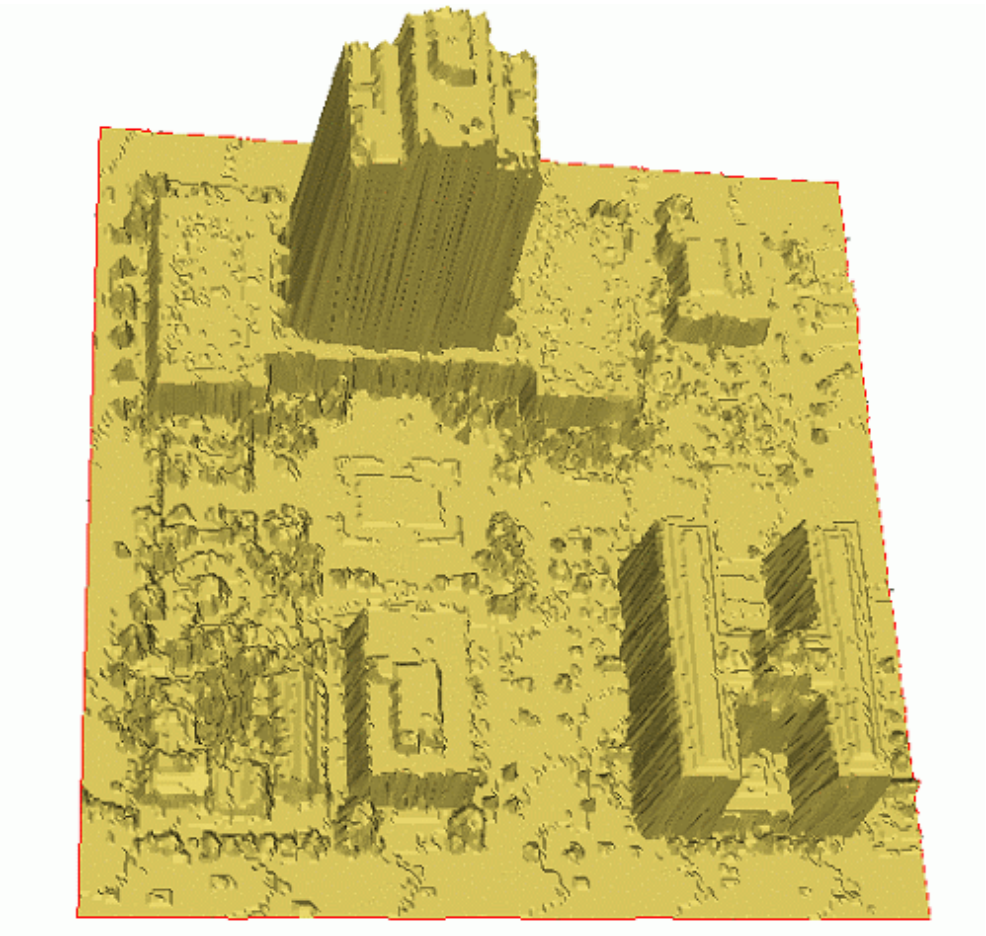

Figure 3.5 Mesh Decimation: Visualization of TIN using VTK

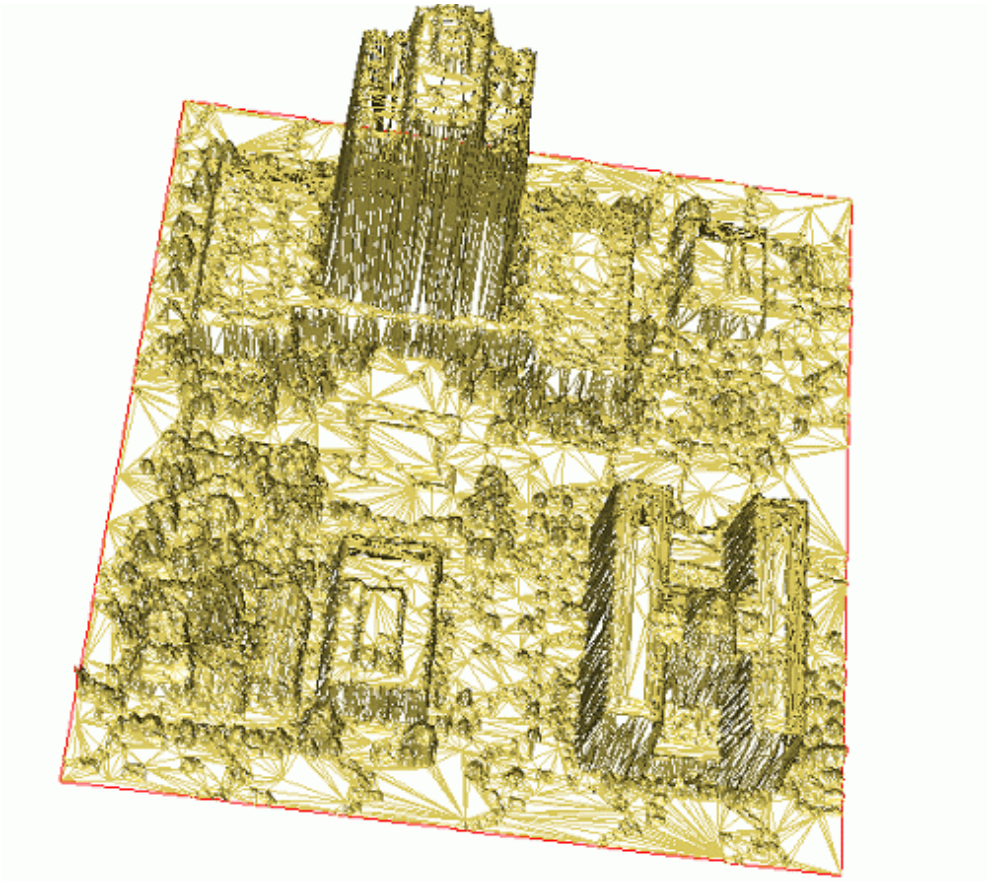

Figure 3.6 Mesh Decimation: Wireframe Model of Figure 3.5

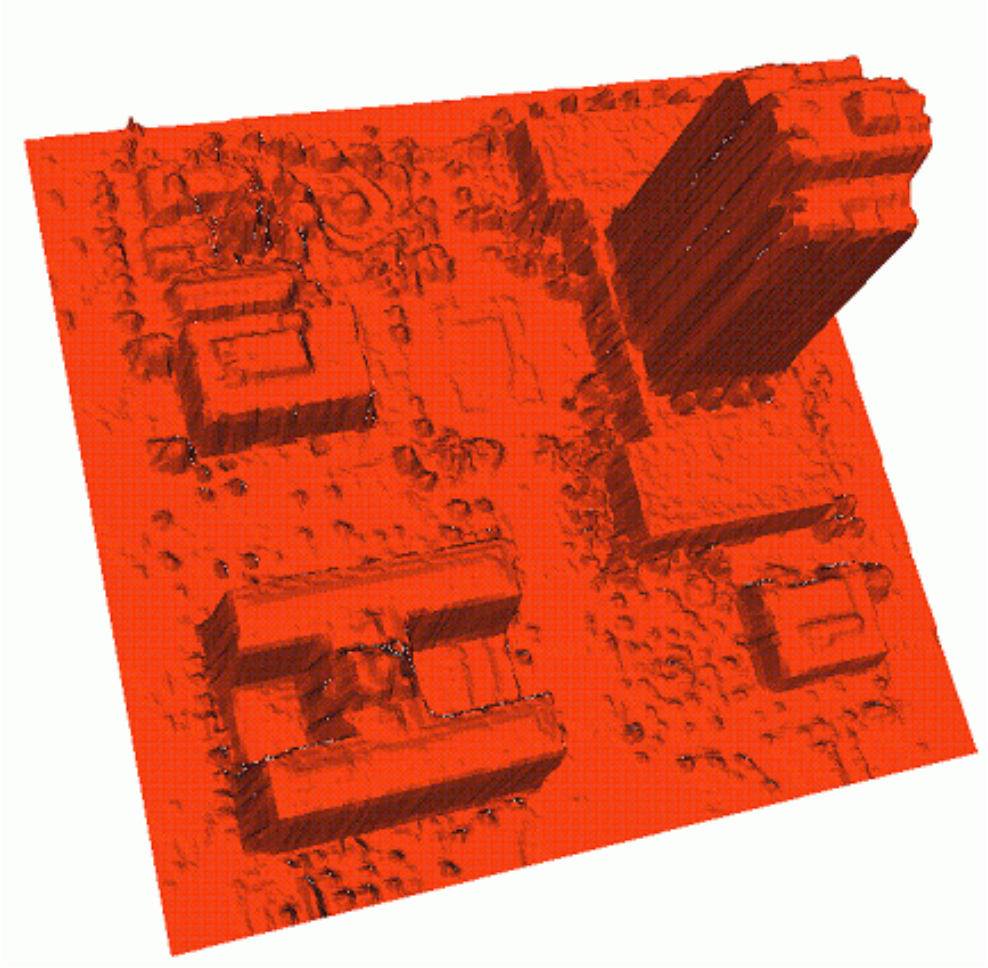

Figure 3.7 Surface Fitting: Visualization using GGTK

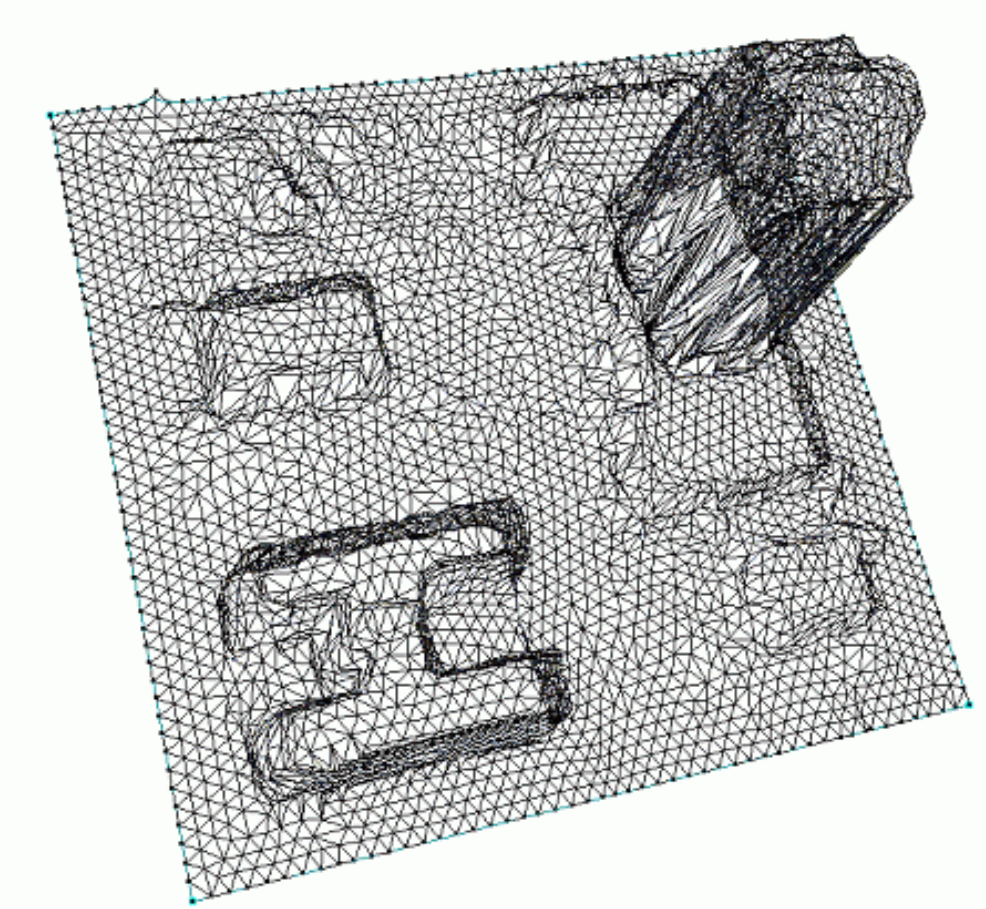

Figure 3.8 Surface Fitting: Wireframe Model of Figure 3.7

Table 3.1 A Summary of Methods used in Initial Experimentation

Method	**Advantage**	**Limitation (w.r.t.DEM image processing)**
Mesh refinement through edge detection using Sobel operators and visualization using TERRA	generates good approximations of terrain data	limited vision of sobel operators fails to reduce noise levels
Mesh decimation by using VTK	good approximations are generated at higher levels of detail	number of points in resultant dataset is high and lower levels of detail result in relaxing topology
Surface fitting using NURBS, visualization using SOLIDMESH	grid points used as control points and hence resultant approximations are very good	number of control points is same as number of original grid points and hence accurate image capture is achieved but no data reduction is achieved.

A synergy between the two techniques could prove to be an arguably better solution.

The next chapter explores a prototype model catering to a probable solution.

CHAPTER IV

AN ALTERNATE APPROACH

The problem of accurately capturing the urban structures from its elevation model not only involves capturing the conforming bounding box of each object but also eliminating the irrelevant entities/objects called 'noise' (refer chapter one) that may be present in the dataset, thereby 'cleaning' the geometry for use in simulations. Since, capturing and segregation relies on the generation of a good filtering mechanism, the problem space can be divided into :

1. generating an appropriate filter to capture relevant entities/objects
2. segregating the relevant objects identified in the DEM image
3. striking a synergy between speed of operation and resolution of final approximation

By employing an iterative development approach towards this problem (see figure 4.1), a multi-pass scan line algorithm is being hypothesized. The urban representation model rests on generating various levels of approximation through the use of filters computed on the basis of a statistical model of the given dataset.

4.1 A Filter Based on the Standard Deviation of the Elevation Data

Generating an appropriate filter is the most important and crucial step in image processing. The choice of the filter determines the quality of the final product and hence care

must be taken to select the most appropriate filter for the problem at hand. Numerous types of filters are available in literature and some of them have been described in the previous chapters, such as Sobel operators, 2x2 filters, 3x3 filters, etc. However, most of these filters were applied to DEM for terrain datasets and hence the resultant approximations were fairly good. But when the same filters are applied to DEM for an urban environment, the approximations revealed a poor attempt at noise elimination and edge capture.

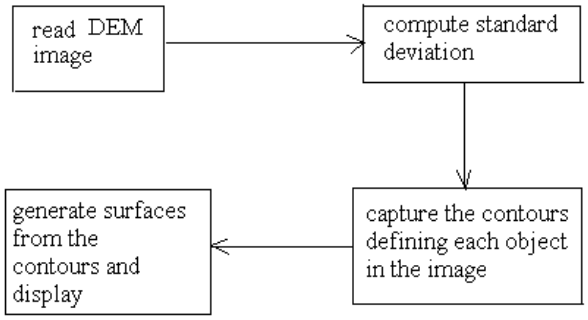

Figure 4.1 Development cycle (Conceptual design model)

An effective filter must be based on the distribution of the elevation at various points on the DEM dataset without having to study the given dataset and analyze it manually. One approach is to simply use the average value of the elevation, but given the urban dataset where the entities are tightly grouped together and may also contain a lot of noise, this technique is not very reliable. A more reliable method applying the concepts of standard deviation to define filtering threshholds is hence used.

Standard deviation is a measure of how tightly the other entities cluster around the mean entity. It is calculated as

$$s_d = \sqrt{\sum_{i<n}(x_i - x_d)^2}/(n-1) \tag{4.1}$$

where x_d is the average elevation value, x_i are individual elevation points and n is the number of points. This computation has been used for determining the spatial organization of the elevation information for the given dataset. Figure 4.2 shows, for example, a distribution of the elevation alues and the standard deviation on either side of the mean of the distribution.

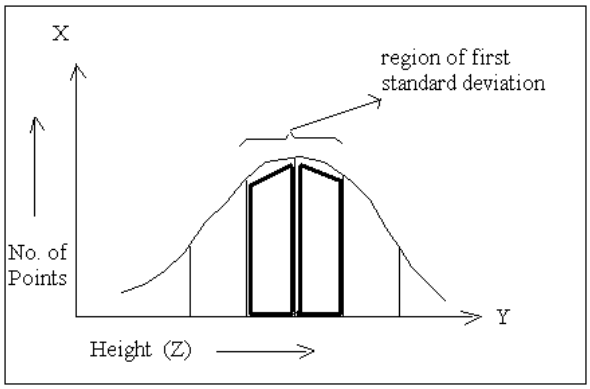

Figure 4.2 Standard deviation

Thresholds or filters are computed using the first standard deviation corresponding to the region in the immediate vicinity of the mean value. Referring to figure 4.2, the first standard deviation is given by the highlighted region. This region is uniformly sliced

to generate the filters. The number of slices is user controlled and it also represents the level of detail that would be generated. For example, an LOD value of four implies that the first standard deviation region is sliced uniformly to give four threshhold or filter values. The remaining region under the curve is sliced at lowest elevation and at the highest elevation to give two more filters. These filters form the cut-off for generating contours which is explained in the next section.

4.2 Collapsed Contouring

Filters defined through standard deviation form the threshold levels that define the cut-off for generating contours capturing the buildings. Unlike the marching cubes algorithm [17], which cuts through the imaginary edges in the DEM grid thus introducing extra points in the process, the modified approach includes only the 'qualifying' point in the contour. A point is said to be 'qualified' if its value (height value) is greater than the present threshhold level and has not been traversed earlier. It may be noted that each point on the DEM grid has four neighbours, viz. north, south, east and west neighbours. Contour generation begins at the first 'qualifying' point encountered and then moves positively or negatively along X or Y axis, (i.e., in the north, south, east or west direction from the point under consideration) until it traverses around the object to complete the contour. Since the traversal is restricted to the points, this type of contouring is distinguished from the traditional understanding by calling it collapsed contouring.

Consider figure 4.3 as a guiding example.The DEM grid of 77 points is processed in the following manner.

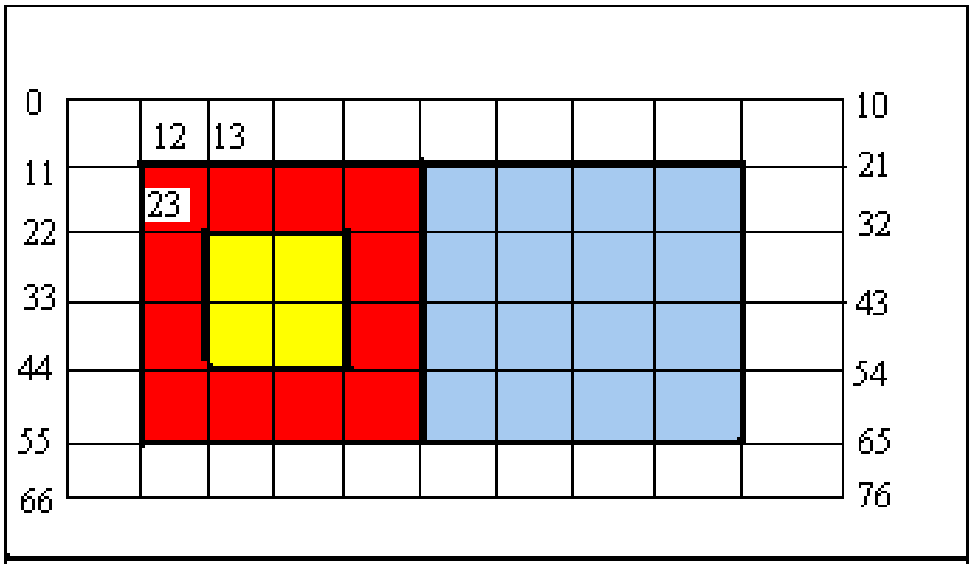

Figure 4.3 Sample dataset of 11 x 7 points Top view

The algorithm begins with a search for the qualifying point. Assume the first threshold level is defined at height 15 feet. Starting the scan through the entire dataset from point 0, we hit upon the qualifying point at the 12th grid point. This point is qualified for consideration because its height value is 40 feet. The contour generation has now been triggered. Examining its neighbouring points, we note that the north neighbour (grid point number equal to 1) is below threshhold at 0. Similarly the west neighbour (grid point number 11) is also below the present threshhold of 15 feet. However, grid points 13 and 23 which form the east and south neighbours respectively, are of elevation

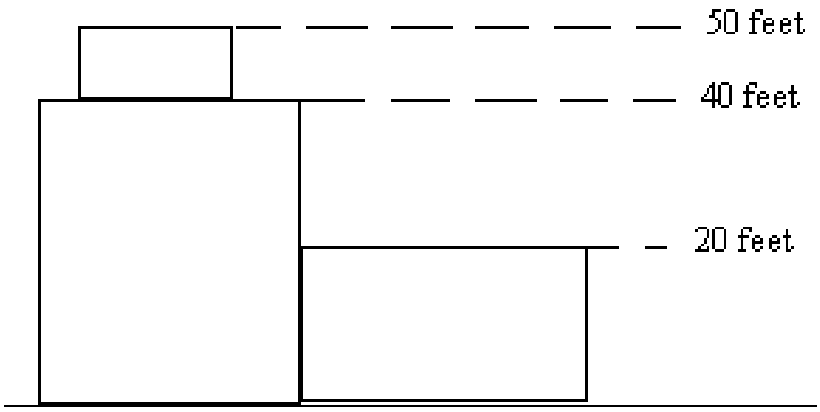

Figure 4.4 Sample dataset of 11 x 7 points Front view

40 feet. Hence the direction of traversal for the contour would be either east or south. By maintaining a table of traversal routes for each direction, the appropriate grid point is selected (see figure 4.5).

The design model for collapsed contouring involves the definition of the 'node' data structure (see figure 4.6). When a point in the DEM grid that is above the particular threshold level is encountered it is stored in 'node'. The connectivity information of this point with its north, south, east and west neighbors, that are also qualified, is stored in the bitset member of 'node'. Bitset is a C++ standard template library data type and is used to determine the direction of traversal when generating contours. Thus, points 12 and 23 which form the east and south neighbors of point 12 are stored in bitset, while points 1 (north neighbor) and 11(west neighbor) are not stored. This process is continued till

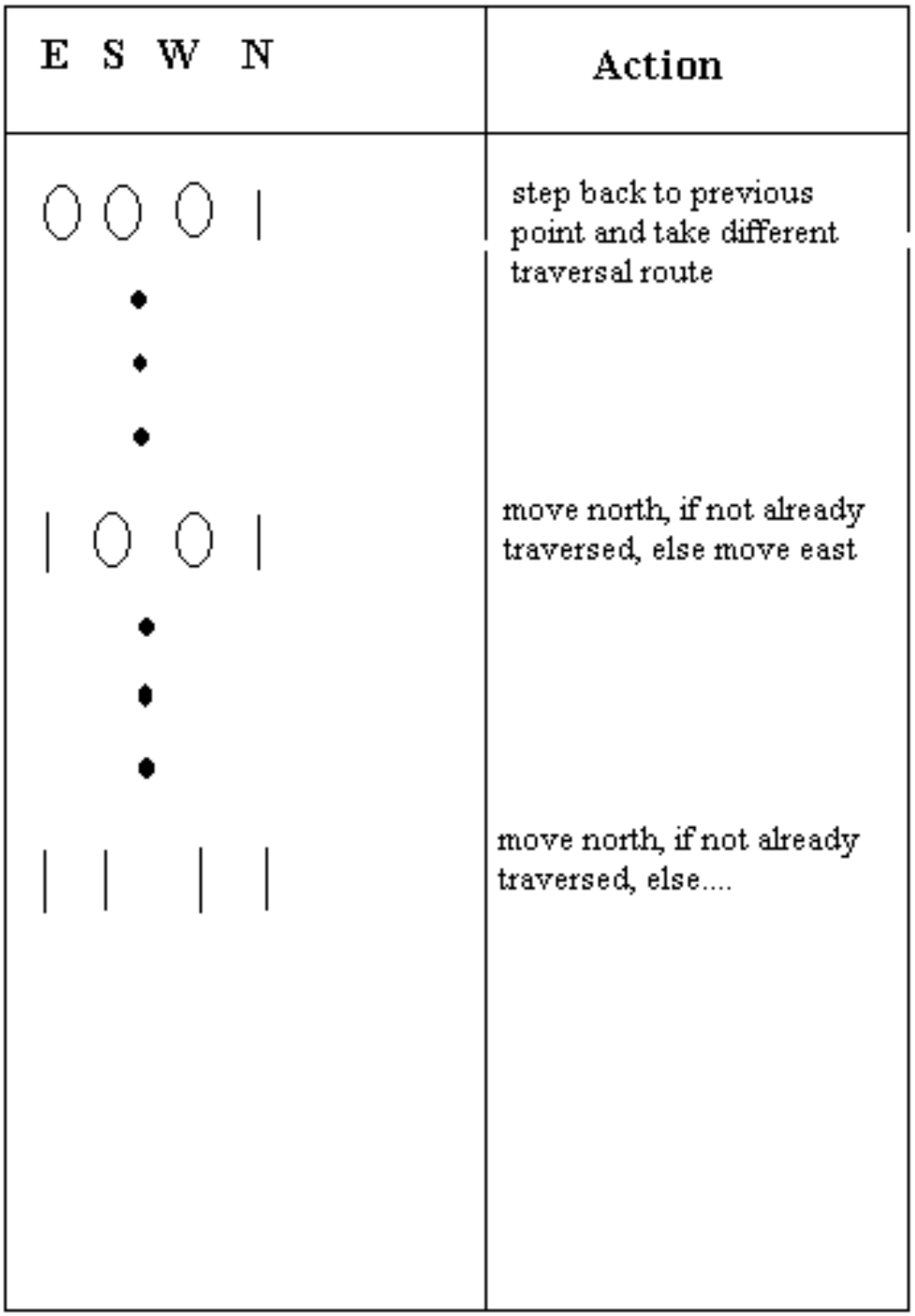

Figure 4.5 Map of Traversal Routes for North, South, East and West Directions

the contour traverses around and returns to the start point 12. Once the entire contour has been captured for the particular threshhold, it is stored in the ’building’ class (see figure 4.6). While ’node’ is the building block of the algorithm ,’building’ forms the tree structure for hierarchical representation. Referring to the example figure 4.4, (0, 10, 76, 66) forms the root contour. Points 12,13, 14, 15, 16, 17, 18, 19,20, 31, 42, 53, 64, 63, 62, 61,60, 59, 58, 57, 56 together form the next contour which is tagged as child to the root and also forms the input boundary for the next iteration. If the next threshold level is at 25, then the dark-shaded region is captured and added as child to the previous contour. Thus, a recursive approach towards this process helps in generating a tree structure with leaf nodes representing building tops and each traversal route gives an entire building. The recursive algorithm mentioned below shows that the bounding box of the most recently generated contour forms the boundary within which the next set of contours is determined.

```
maketree (Xmin, Xmax, Ymin, Ymax) {
if(threshhold == MaxThreshholdLevel)
  return;
else {
 for(i=Xmin, i<Xmax; i++)
 for(j=Ymin; j<Ymax; j++)
 if(grid[i][j] <= threshhold) {
  list = generateContour(i, j);
  Building *child = addchild(parent);
  setChildren(parent);
  setContour(list);
  getBoundingBox(list);
  maketree(newXmin, new Xmax, newYmin, newYmax);
}
}
}
```

```
generateContour (i, j) {
while (newNode != startNode) {
valency = getNeighbours(i, j);
switch(valency) {
  case 1: ...
  case 2: ...
  .
  . -----> returns a point that forms a link in the contour.
  .
 }
 }
 }
```

This approach helps in generating various levels of approximation for the given dataset. The topmost levels represent the ground levels. This unique approach towards the problem also handles regions of interest where the ground level is not even or is slightly hilly. More contours, which improve the quality of the approximation, can be generated by setting higher values for the level-of-detail (LOD) control. The quality is also dependent upon the amount of noise that has been safely removed from the original image dataset. As explained in the first chapter, noise elimination is an important task. Noise due to high frequency elevation, such as towers and lamp-posts, is addressed by the standard deviation function explained in the previous section while the noise in the x-y direction, viz. cars, bushes, etc., is addressed by a ’noise controller’.

A noise controller is a simple conditional statement that disregards contours of very small size. It controls the ’noise factor’ (NF) and it regulates the minimum number of points that may be present in any contour at any level. For example, a noise factor of 20 implies that any contour which has total number of grid points less than 20 can be

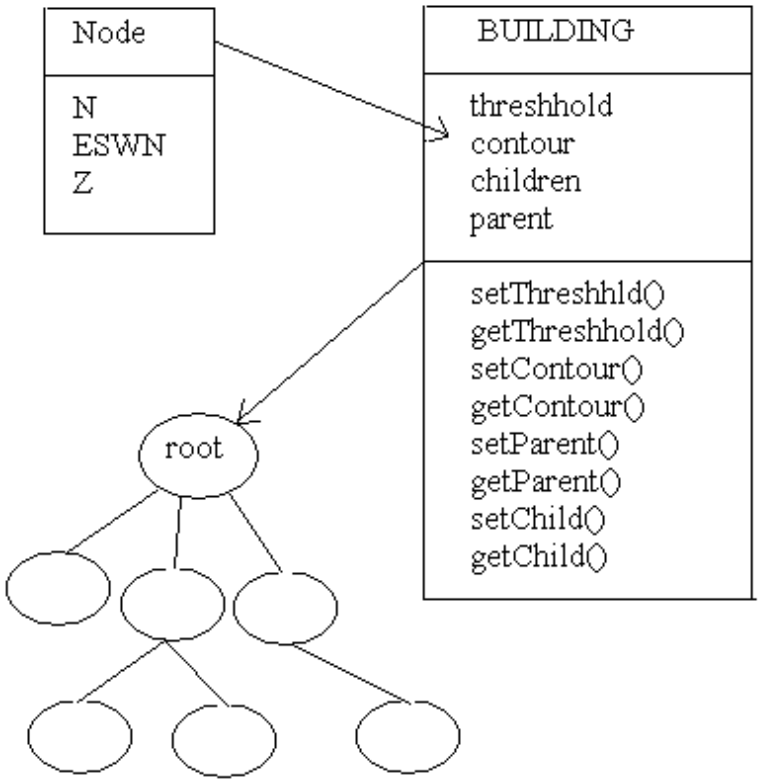

Figure 4.6 Building blocks of the algorithm

safely eliminated. Optimal values for LOD and NF are determined by performing a simple comparison of trial values. Chapter 5 describes the use of NF and LODs through various examples.

While collapsed contouring captures the shapes of the buildings accurately, it may be possible to further reduce the number of points in each contour by eliminating the ones that lie along the same line. This is called contour compression and, simply put, it involves representing a straight line by two points and eliminating any other redundant point between the end points. So the final elements in the contour for our example dataset would be 12, 20, 64, and 56. Thus a 25-points contour is now reduced to four points.

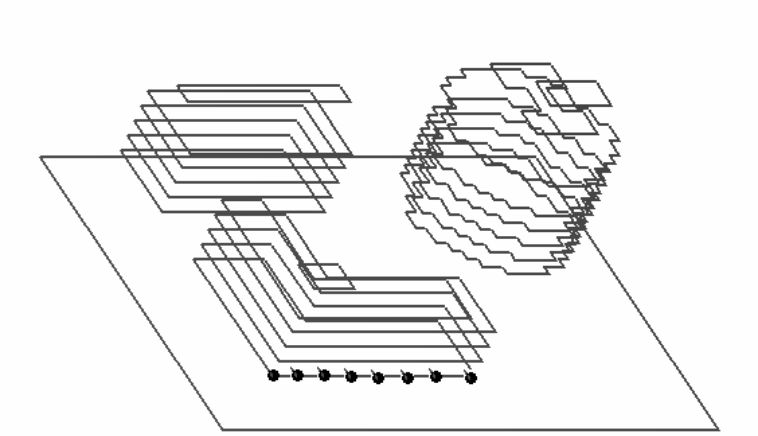

Figure 4.7 Contours captured in a 50 x 36 DEM grid

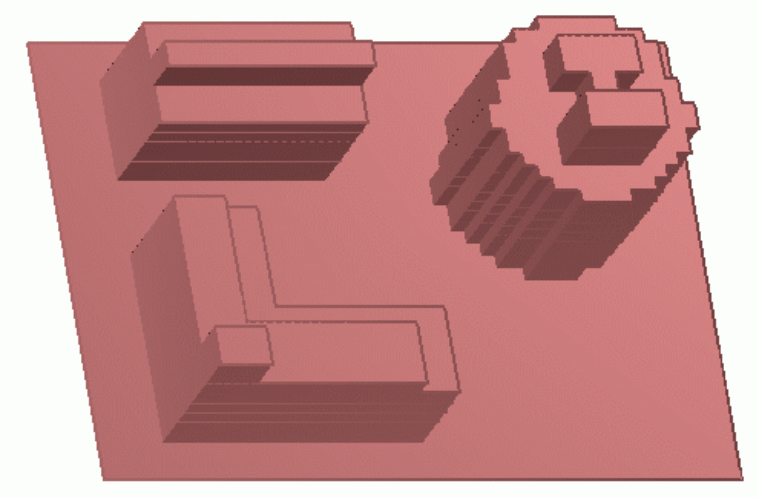

Figure 4.8 Surface representaiton of figure 4.7

4.3 Summary

A complete quantitative and qualitative analysis of the resultant approximations is documented in the next chapter. However, it is noteworthy to mention that that this prototype model is able to capture the entities in the Salt Lake city dataset in an efficient manner and the various levels of detail generated provide the right synergy between resolution of approximations and thier size. As shown in figures 4.7 and 4.8, the complex shapes of the buildings (circular, h-shaped, L-shaped etc) in a manually created 50 x 36 point DEM grid have been captured and contours are drawn at various threshhold levels. Figure 4.7 also shows the un-compressed version of the contour and hence the extra points on the straight edge. These redundant points are eliminated through the contour compression routine explained earlier. Unlike the algorithms studied in Chap-

ter 2, this alternate approach does not introduce any additional points into the dataset through interpolation (as in the case of Marching Cubes algorithm) and nor does it fail to avoid high frequency noise (as in case of limited vision of 2 x 2 or 3 x 3 filters used in generating TIN models). For display purposes, the entire resultant approximation is triangulated using a simple triangulation technique. The sets of triangles are rendered using the OpenGL library routines. The next chapter documents the results obtained from this alternate model when applied to real-world test cases. It also summarizes some of the observations made and states the advantages and limitations of this method.

CHAPTER V

RESULTS AND EVALUATION

The results obtained from investigating the alternate approach discussed in the previous chapter are discussed in this chapter. A quantitative evaluation of the algorithm helps us understand the features of this particular model and also the application areas to which it could be applied. A qualitative evaluation of the same provides insight into the complexity of the algorithm and its running time. The analysis and evaluation phase is organized into three components:

1. a comparison of the resultant approximation with the original image in terms of accuracy of capture, i.e., the extent of preservation of certain shapes and statistics of the objects encountered
2. a comparison of the new approach with existing techniques
3. an analysis of complexity of algorithm

5.1 Accuracy of Image Capture for the New Approach

This research uses object dimensions to measure the accuracy of the approximation of the actual object. The accuracy measure indicates the preciseness of the algorithm in capturing the exact shape of the objects encountered in the DEM image.

As a test case, the exact real-world dimensions of a certain H-shaped building (the Utah Hotel) was used (see figure 5.1a and 5.1b). The Utah Hotel building stands approx-

imately 10 storeys tall and is surmounted by a four-storey high tower. The area of the building is 176 x 200 square feet. By enabling the user to set the LOD and NF values, a trade-off between image size and quality can be obtained.

The dataset for the Utah Hotel was extracted from the Salt Lake City dataset and it consists of a rectangular grid of 70 x 85 points. Using a prototype, the new approach algorithm was executed for two sets of input values. Figure 5.2(a) represents the approximate surface obtained with level-of-detail value being equal to 1 and noise factor (NF) of 30 and Figure 5.2(b) represents the approximation for LOD=4, and NF=5.

Figure 5.1 Utah Hotel (176 x 200 x 185 cubic feet)

Table 5.1 shows the results of the test case execution.

The dimensions correspond to the base of the H-shaped building, which means the lowest contour capturing the H-shape. The results indicate a fairly good approximation. The algorithm captures the shape of the building accurately, but the exact statistics are

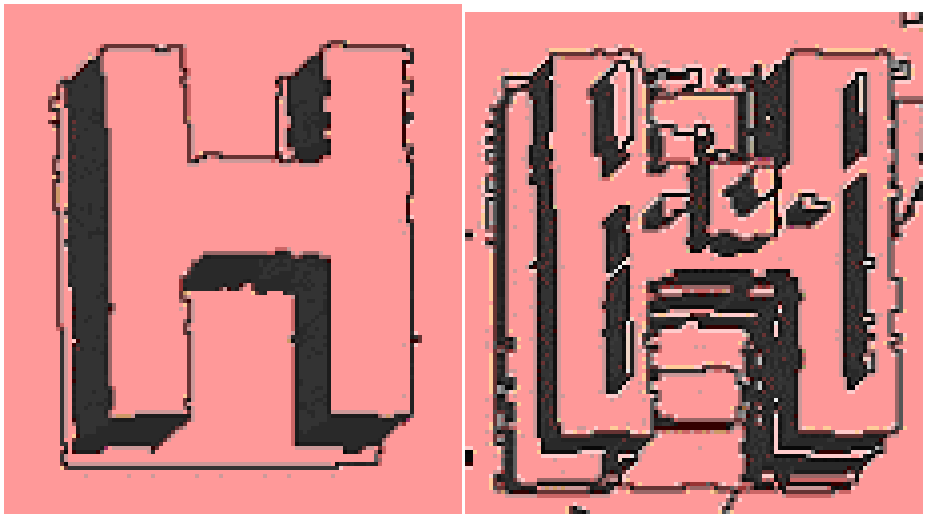

Figure 5.2 Approximations of Utah Hotel. (a) LOD=1, NF=30, (b) LOD= 4, NF=5

Table 5.1 Comparison chart of Utah hotel building statistics

Utah Hotel	**Number of Points**	**Width (feet)**	**Length (feet)**	**Height (feet)**	**LOD**	**Noise Factor**
Original	5950	176	200	185	-	-
Case I	139	200	258	135.1	1	30
Case II	584	200	232	165.2	4	5

not maintained. The higher difference between length and width of the original image and its approximation can be partly attributed to the presence of bushes around the building. So if the height of the bushes and tress is above the current threshold level and they lie in close proximity to the building (resolution of the dataset is one meter), then the contouring algorithm would also include these points too.

5.2 The New Approach Compared to Earlier Experimentation

As discussed in Chapter 3, numerous experiments were conducted using preprocessors to existing software packages or by using the software directly. One of the techniques described was the decimation approach towards approximation (see section 3.1.2). A comparison of the results on a 242 x 249 grid of the original Salt Lake City DEM dataset using the previous approaches and the new technique is used here to understand the merits and demerits of each approach. A visual evaluation of the two resultant approximations is based on :

1. the preciseness in capturing the shape of the buildings, and
2. the extent of reduction in noise levels.

A more concrete comparison involves the determining the number of points used to represent the particular approximation. Figures 5.3 (same as figure 3.5, reproduced for convenience) shows the original DEM image visualized using VTK. It contains 60,258 points. Figure 5.4 (same as figure 3.6, reproduced for convenience) is the approximate surface obtained after performing mesh decimation using VTK. The number of points

was reduced to 25,621 points. Any further reduction in the number of points would involve relaxing the topology.

Figures 5.5, 5.6 and 5.7 depict the results of image processing using the alternate approach for different levels of detail and noise factor. While figure 5.5 does not capture all the relevant entities in the dataset due to the presence of varying frequency noise, the number of points has been reduced drastically from 60,258 points to a mere 623 points. Such an approximation could be used in realtime and interactive applications while time is an important factor. Figure 5.6, with LOD of 4 and noise factor of 40, provides a better approximation while keeping the number of points still relatively low at 2,642 points. It may be noted that while the noise factor is the same in figures 5.5 and 5.6, the increase in the LOD factor contributes to a more detailed approximation. Similarly, figure 5.7, with LOD=4 and NF=10 provides a still better approximation because with an NF of 10, even the smaller objects have been captured. Contrasting the results of this case with that obtained using VTK, we see that the number of points is still very low at 4,248 points as opposed to 25,621 points for the similar visual results.

Table 5.2 gives a comparison chart for the two techniques.

5.3 Algorithm Complexity for the New Approach

The time complexity of the algorithm revolves round three basic operations:

1. computing filters or threshold levels using standard deviation,
2. generating contours for each threshold level, and

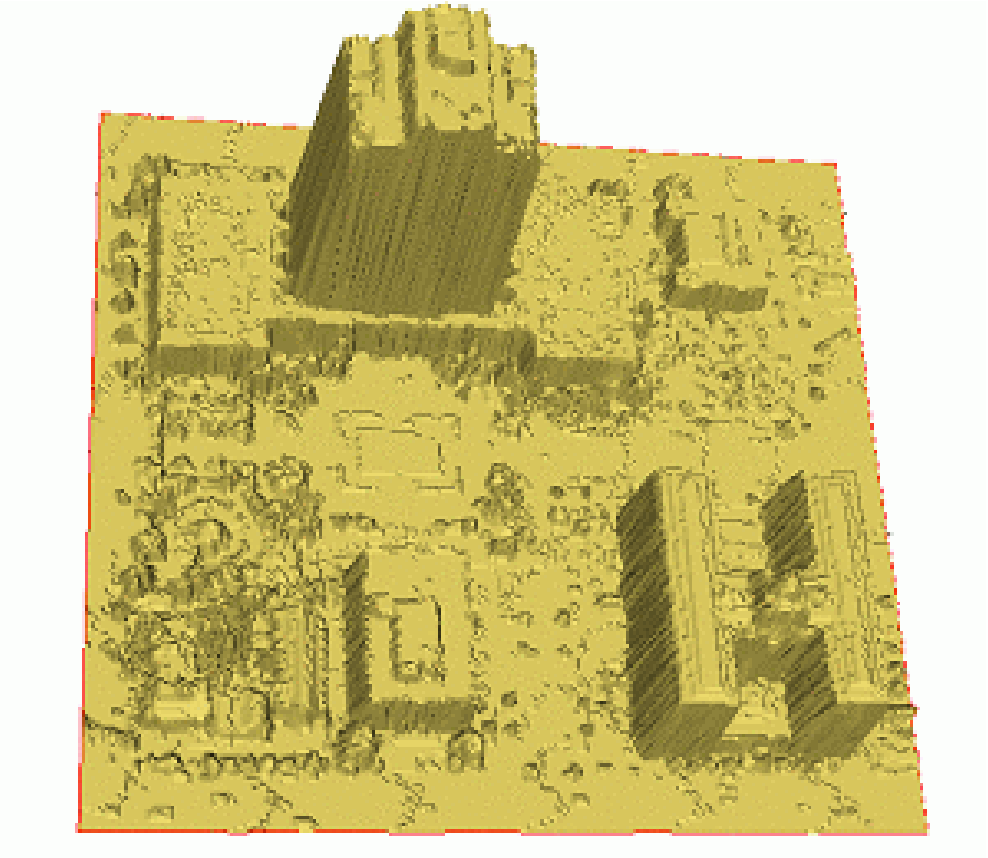

Figure 5.3 Visualization of Main Square DEM image using VTK (242 x 249 points)

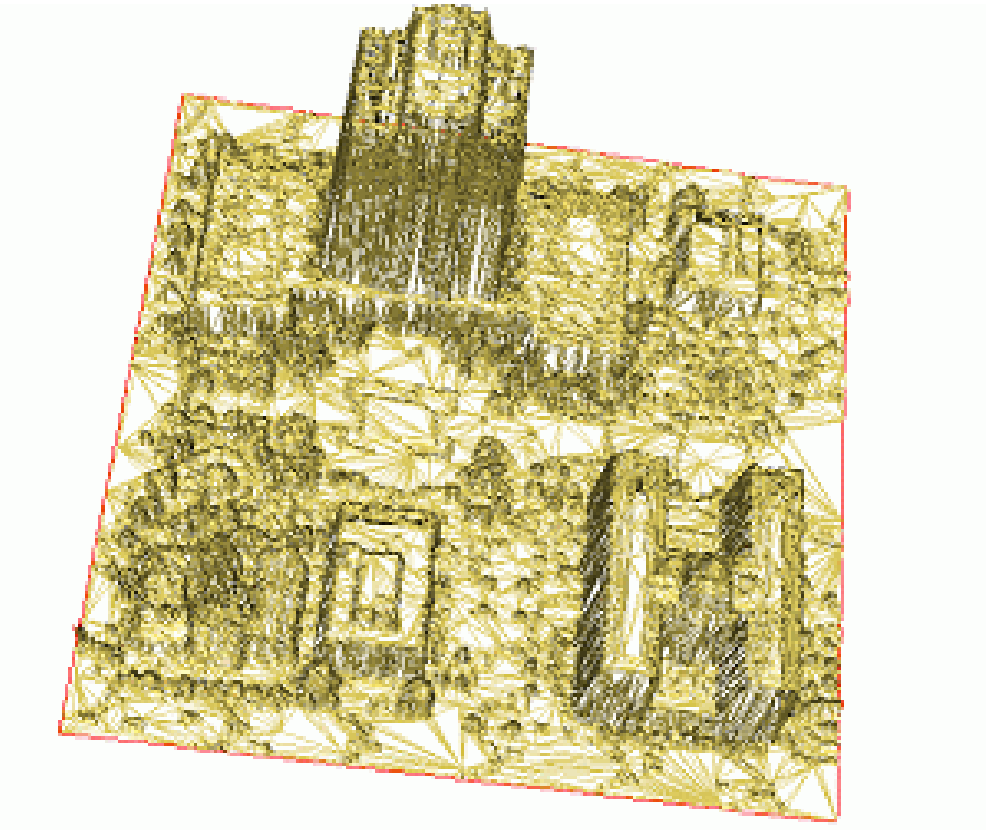

Figure 5.4 Main Square approximation using VTK mesh decimation

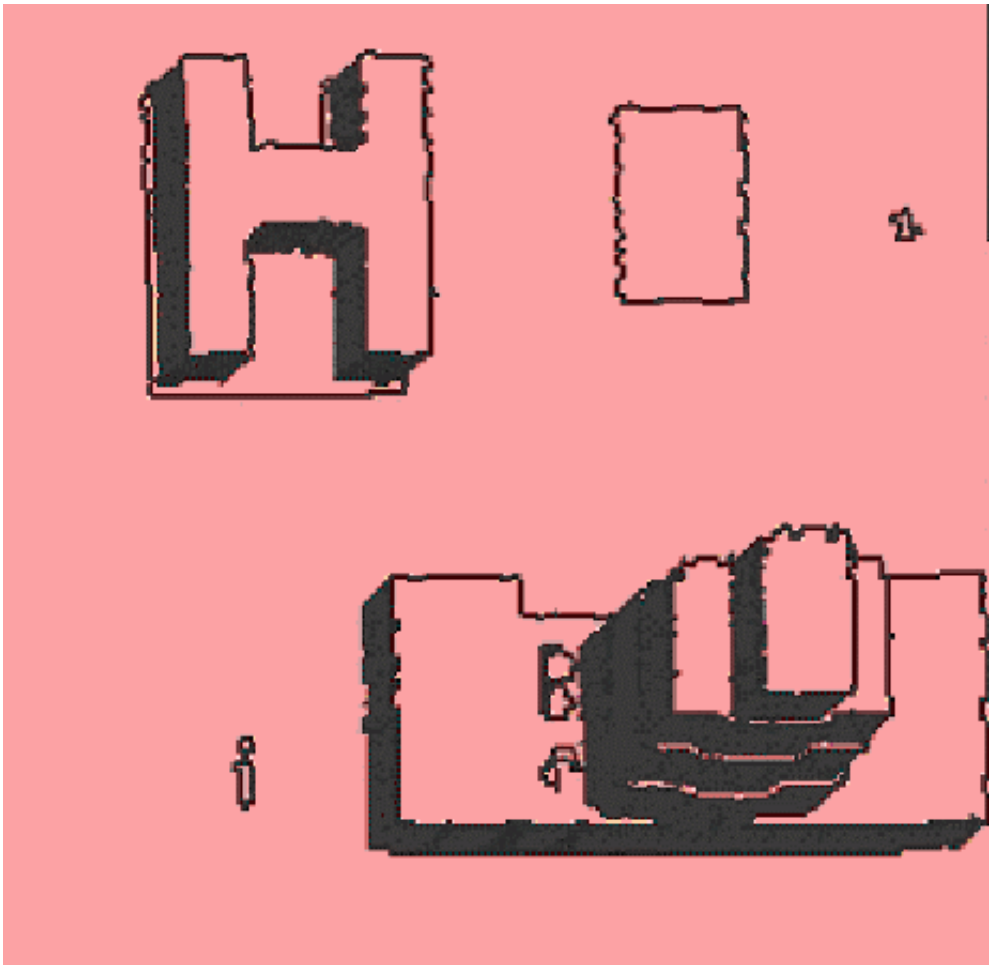

Figure 5.5 Main Square approximation using new model (623 points, LOD=1, NF=40)

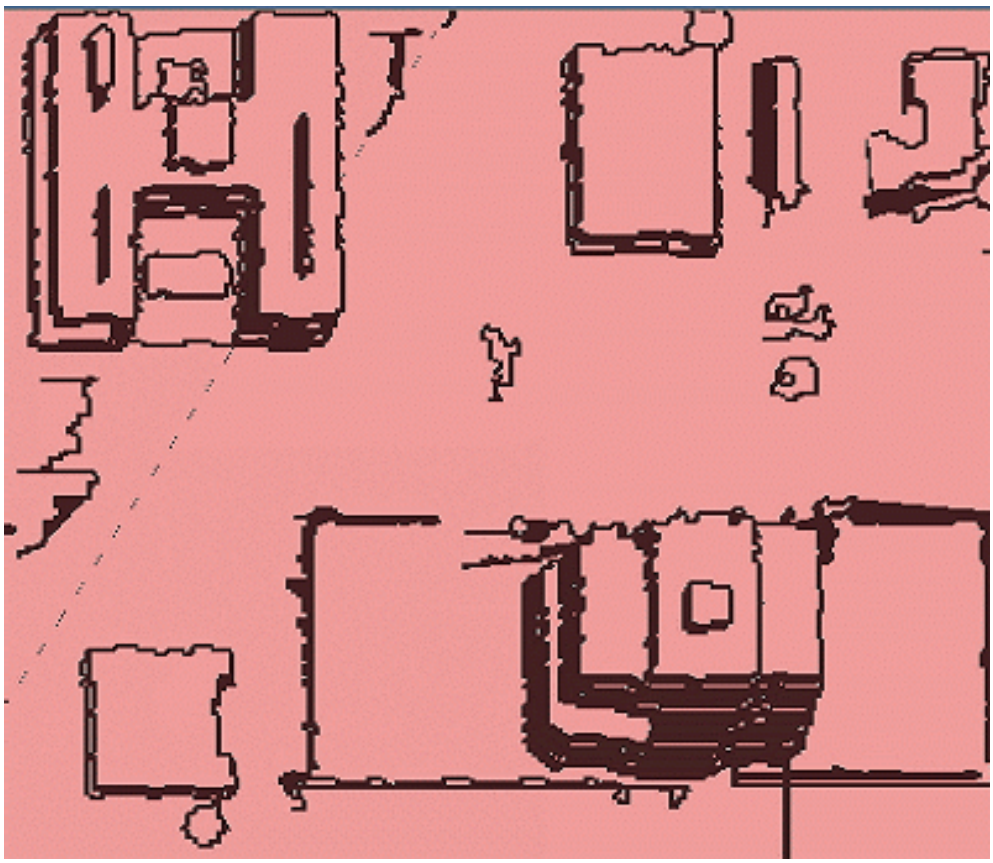

Figure 5.6 Main Square approximation using new model (2,642 points, LOD=4, NF=40)

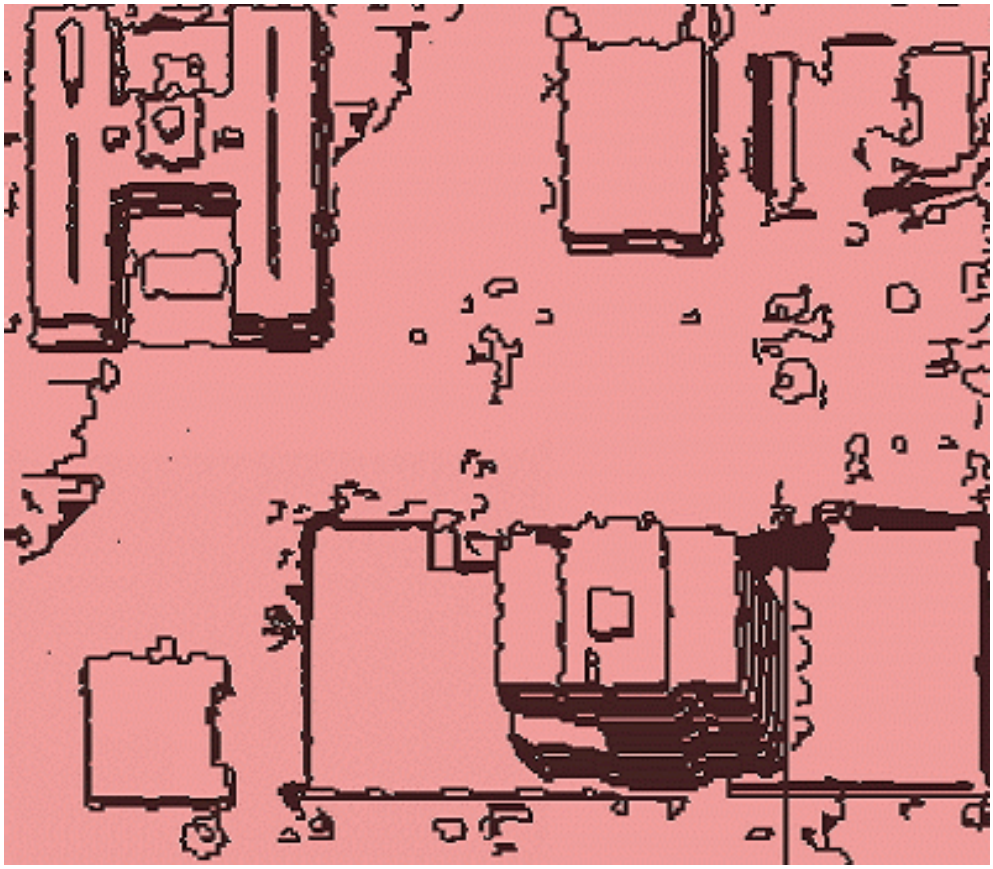

Figure 5.7 Main Square approximation using new model (4,248 points, LOD=4, NF=10)

Table 5.2 Comparison chart for mesh decimation versus prototype model

Algorithm	**Original Size**	**Number of Points in approximation**	**LOD**	**Noise Factor**
Mesh decimation using VTK	60,258	25,621	-	-
Prototype model (Case I)	60,258	623	1	40
Prototype model (Case II)	60,258	2,642	4	40
Prototype model (Case II)	60,258	4,248	4	10

3. building the tree structure depicting hierarchical organization of the buildings or entities in the dataset.

The computation of the standard deviation involves addition, subtraction, multiplication and division (see figure 4.3). Each of these operations take constant time and hence, for a dataset of size n, it can be performed in O(n) time. The contouring routine is dependent upon switch statements, which are again executed in constant time and hence can be performed in O(n) time.

Building the tree structure involves recurrence and the subdivision at each level can vary. It is not a binary tree or a quad tree or an octree. The best case scenario would involve a single division which is highly unlikely as the possibility of more than one object being present in a given DEM set is very high. Considering the worst case sce-

nario of ’m’ subdivisions, the time complexity of this operation is calculated to be O(m lgmn).

Hence the total time complexity of the algorithm is O(n + mlgmn).

5.4 Summary

So, as hypothesized, the prototype model proves to be a good image processing and data reduction tool for urban environment simulations. As is seen from the results, the algorithm is able to capture the edges of the building reliably because it does not interpolate between edges and is also able to eliminate considerable amounts of noise to give a clean geometry. The next chapter summarizes the results and lists the merits and some of the limitations of the model. It also outlines the scope for future work in this direction.

CHAPTER VI

CONCLUSION

The aim of this thesis was to explore an alternate approach to image processing and data reduction of DEM data for urban environments. A study of the existing software and surface approximation techniques revealed a high dependence upon the use of 2x2 or 3x3 filters to decide which points are to be included in the approximation. This limits the vision of the filter and hence the noise levels are still relatively high. Also, the edges of the buildings were not accurately captured.

But the results obtained from a prototype using the new approach proposed here clearly indicate the efficiency of the algorithm in capturing the edge information of the buildings as well as maintaning the noise levels at a minimum. The use of the standard deviation to generate filters helps in understanding the spatial distribution of the elevation information. Also, contours formed by selecting from existing points (as opposed to those obtained by interpolating between points) helps in reducing the size of the resultant dataset. By manipulating the LOD and noise factor controls, a relatively accurate and clean geometry of the DEM data can be obtained.

So summarizing:

1. A simple and economical model has been developed using an alternate approach to the problem of DEM data reduction and processing.
2. Approximations derived from the new model represent a clean geometry.

3. The alternate approach is capable of capturing the relevant entities in the DEM image accurately.

4. The new model is capable of capturing buildings resting on uneven ground levels at higher LODs.

5. The new model can represent complex entities using a smaller set of points, so while VTK requires approximately 25,000 points to approximate a 60,000-point DEM grid the prototype requires only 4,200 points and still maintains the topology.

6. The trade-off between quality of resultant approximation and it size can be maintained by manipulating the LOD and NF controls of the new model.

7. The complexity of the algorithm for the new model is also reasonable at O(n+mlogmn), where 'n' is the total number of poins in the original DEM grid and 'm' represents the maximum number of children a node has in the tree structure.

However, the new model has some limitations:

1. The accuracy of the new algorithm is limited by the resolution of the original DEM. As seen from the table 5.1, difference between the statistics of the H-shaped building and those of its approximation were a result of this limitation. Hence, any point which is within this resolution will be considered by the algorithm and included in the contour.

2. The noise factor control eliminates contours simply on the basis of the number of points that it contains. This means that a very narrow and long stretch of pine trees would still be included as a contour if it does not satisfy the elimination criteria. This may not be acceptable. A better alternative might be the use of 'area' of the contour as an elimination criteria.

3. Circular buildings and sloped building edges are represented by straight lines. When traversing around an entity, the algorthm includes only those points in the contour which are qualified and does not interpolate between points. Since the points are arranged in a horizontal rectangular grid format, curved objects such a dome would consist of edges aliged at 90 degree angle.Also, hilly areas or uneven ground levels are represented as plateaus for that particular elevation,

4. The resultant approximations cannot be used directly by flow solvers because the aspect ratio of the triangles generated is not very good, so some form of preprocessing by a grid generator is necessary.

5. The model fails to capture the buildings accurately for very large datasets, (of the order of a million points) if the buildings are resting on highly uneven ground levels. The filters generated by the standard deviation technique for such huge datasets seem inadequate.

An immediate improvement in the complexity of the algorithm can be achieved by merging the contour-compression routine with the contour-generation routine. As discussed in Chapter 4, the contours generated by the algorithm were post-processed to eliminate the extra points that may lie between a straight edge. This could be integrated with the contour-generation routine itself. Any redundant point need not be added in the contour list at all. Also, as seen from figure 4.9 and 4.10, circular buildings are represented using straight edges. By using Non-uniform Rational B-splines (NURBS), curved edges and circular buildings may be captured well. This also means that the new model could be extended for use in medical applications such as in MRI or CT image processing and data reduction. The triangulator used by this model is a naive implementation and, hence, the resultant approximations cannot be used directly by the flow solvers of wind field simulators and would need to be processed by a grid generator. A better triangulation algorithm which would also be able to generate trianlges of good aspect ratio would prove to be very beneficial. The resultant approximation from such a triangulator could be used directly by flow solvers or wind field simulators.

REFERENCES

[1] A. Varshney, Hierarchical Geometric Approximations, doctoral dissertation, Department of Computer Science, University of North Carolina, 1994.

[2] Al-Rousan, N. Cheng, P. Petrie, G. Toutin, T. H. Valaden and M. J. Zoej,"Automated DEM Extraction and Orthoimage Generation for SPOT Level 1B Imagery", *Photogrammetric Engineering and Remote Sensing*, vol. 63, no.8, 1997, pp. 965-974.

[3] Bob Fisher, Simon Perkins, Ashley Walker and Erik Wolfart, "Feature Detectors", http://www.cee.hw.ac.uk/hipr/html/featops.html (current Jul 17, 2002).

[4] David Wladis, "Automatic Lineament Detection Using Digital Elevation Models with Second Derivative Filters", *Photogrammetric Engineering and Remote Sensing*, vol. 65, no. 4, 1999, pp. 453-458.

[5] Dominique Durand, "DEM, Stereoscopic Aspects of SPOT", The Science of Remote Sensing,http://ceos.cnes.fr:8100/cdrom-00b2/ceos1/science/gdta/ang/a2an/1.htm (current 17 Jul 2002).

[6] F. J. L. Hulle and J. J. Schuurman, Wind Measurements at Zolder (B), technical report ECN-c–97-010, Energy Research Center of the Netherlands, The Netherlands, 1997.

[7] Gang Chen and Yee H. Hong Yang, "Edge Detection by Regularized Cubic B-Spline Fitting", itshape IEEE Transactions on Systems, Man and Cybernetics, vol. 25, no.4, 1995, pp. 636-643.

[8] H. P. Borouchaki, P. Lafon, P. Laug and P. L. George, "Minimal Variational Surfaces and Quality Meshes", *Proceedings of the Ninth International Meshing Roundtable*, New Orleans, Louisiana, Oct 2000, Sandia National Laboratories, pp. 217-226.

[9] J. R. Sulebak, Applications of Digital Elevation Models, white report, SINTEF Institute of Applied Mathematics, Department of Geographics Information Technology, Norway, 2000.

[10] Joe F. Thompson, Bharat K. Soni and Nigel P. Weatherill, eds., *Handbook of Grid Generation*, Washington, D. C., CRC Press, 1999.

[11] Kim Taejung and Yong-Jo Im, ”Automated DEM Extraction from the Kompsat-EOC Images”, Presented: 22nd Asian Conference on Remote Sensing, Singapore, Nov 2001, Centre for Remote Imaging, Sensing and Processing (CRISP), National University of Singapore. Singapore Institute of Surveyors and Valuers (SISV), Asian Association on Remote Sensing(AARS).

[12] Michael Garland and Paul S. Heckbert, ”Fast Triangular Approximation of Terrains and Height Fields”, http://graphics.cs.uiuc.edu/ garland/papers.html (current 6 Sep 2001).

[13] Michael Garland, ”Terrain Simplification”, http://graphics.cs.uiuc.edu/ garland/CMU/scape/ (current 16 Nov 2002).

[14] P. J. C. Brown, Selective Mesh Refinement for Rendering, doctoral dissertation, Emmanuel College, University of Cambridge, UK, Feb 1998.

[15] Paul S. Heckbert and Michael Garland, ”Survey of Polygonal Simplification Algorithms”, http://graphics.cs.uiuc.edu/ garland/papers.html (current 16 Nov 2002).

[16] S. Gopalsamy, R. P. Koumullil, N. S. Doddamani, ”Surface Grid Generation Over a City from DEM Data”, Presented: Fifth Mississippi State Conference on Differential Equations and Computational Simulations, Starkville, Mississippi, May 2001.

[17] W. E. Lorensen and H. E. Cline, ”Marching Cubes: A High Resolution 3D Surface Construction Algorithm”, *Computer Graphics*, vol. 21, July, 1987, pp. 163-169.

[18] William J. Schroeder, Jonathan A. Zarge and William E. Lorensen, ”Decimation of Triangle Meshes”, *Computer Graphics*, vol. 26, July, 1992, pp. 65-69.

[19] William J. Schroeder, William E. Lorensen and Ken Martin, ”The Visualization Toolkit”, http://public.kitware.com/VTK (current 16 Nov 2002)

[20] Y. Huang and J. C. Trinder, ”Object Recognition Based on Boundary Description”, *Photogrammetric Engineering and Remote Sensing*, vol 65, no. 8, 1999, pp. 915-921.