Wavelet-based Multiresolution Data Representations for Scalable Distributed GIS Services

by

Jingsong Wu

B.S., Wuhan University of Hydraulic and Electric Engineering, China (1993)
S.M., Tsinghua University, China (1996)
S.M., Georgia Institute of Technology (1997)

Submitted to the Department of Civil and Environmental Engineering
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in Computational Engineering and Information Technology

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2002

Author .....
Department of Civil and Environmental Engineering
May 16, 2002

Certified by.....
Kevin Amaratunga
Assistant Professor
Thesis Supervisor

Accepted by .
Oral Buyukozturk
Chairman, Department Committee on Graduate Students

Wavelet-based Multiresolution Data Representations for Scalable Distributed GIS Services

by

Jingsong Wu

Submitted to the Department of Civil and Environmental Engineering
on May 16, 2002, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computational Engineering and Information Technology

Abstract

Demand for providing scalable distributed GIS services has been growing greatly as the Internet continues to boom. However, currently available data representations for these services are limited by a deficiency of scalability in data formats.

In this research, four types of multiresolution data representations based on wavelet theories have been put forward. The designed Wavelet Image (WImg) data format helps us to achieve dynamic zooming and panning of compressed image maps in a prototype GIS viewer. The Wavelet Digital Elevation Model (WDEM) format is developed to deal with cell-based surface data. A WDEM is better than a raster pyramid in that a WDEM provides a non-redundant multiresolution representation. The Wavelet Arc (WArc) format is developed for decomposing curves into a multiresolution format through the lifting scheme. The Wavelet Triangulated Irregular Network (WTIN) format is developed to process general terrain surfaces based on the second generation wavelet theory. By designing a strategy to resample a terrain surface at subdivision points through the modified Butterfly scheme, we achieve the result: only one wavelet coefficient needs to be stored for each point in the final representation. In contrast to this result, three wavelet coefficients need to be stored for each point in a general 3D object wavelet-based representation. Our scheme is an interpolation scheme and has much better performance than the Hat wavelet filter on a surface. Boundary filters are designed to make the representation consistent with the rectangular boundary constraint. We use a multi-linked list and a quadtree array as the data structures for computing. A method to convert a high resolution DEM to a WTIN is also provided.

These four wavelet-based representations provide consistent and efficient multiresolution formats for online GIS. This makes scalable distributed GIS services more efficient and implementable.

Thesis Supervisor: Kevin Amaratunga
Title: Assistant Professor

Acknowledgments

I am very grateful to my advisor, Professor Kevin Amaratunga, who led me into the wavelet world and started me on this research. During my four-year stay at MIT, he has given me tremendous and invaluable guidance, help, and encouragement.

I am also grateful to my thesis committee members, Professors Joseph Ferreira, Jr., John R. Williams, and George Kocur, who have patiently sat through committee meetings and taken the time to understand my work.

I feel fortunate to have many friends among the faculty and students at MIT. They make my life here very enjoyable and intellectual.

My research is supported by a grant from the Suksapattana Foundation (Bangkok, Thailand) to the Intelligent Engineering Systems Laboratory at Massachusetts Institute of Technology. This support is greatly appreciated.

Foremost, I want to thank my father, Xinzheng Wu, my sisters, Feng Wu and Jing Wu, and my brother, Xuesong Wu. They have given me infinite care and support.

In memory of my mother,

Youhong Luo,

who taught me to persist in my goal

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and Scope

A Geographic Information System (GIS) is a computer system that can process geospatial data through various geoprocesses. The earliest work can be traced back to the 1950s [8]. Modern GISs developed in the 1980s. Since then, a lot of commercial software has been developed. Today, because of the rapid improvement of the price/performance ratio of computer hardware, PC-based GIS products have begun to appear in our daily lives. The popularity of the Internet has further provided a great opportunity to host GIS services on the Internet. Several organizations, such as the OpenGIS Consortium (OGC, www.opengis.org) and the National Center for Geographic Information and Analysis (NCGIA, www.ncgia.org), have been formed to provide places for researchers and developers to share ideas and improve research. OGC has put forward an important concept "Open GIS," which requires future GIS products to be interoperable. This means a system should be able to transparently access heterogeneous geodata and geoprocessing resources in a network environment.

Based on interoperability, however, hosting GIS services in a network environment requires good schemes to manage and transfer large amounts of geospatial data. In this research, we construct data models for scalable distributed GIS services. Here, two characteristics of the system are emphasized. One is "distributed," which means the GIS services are network based. This network environment can be Internet based,

wireless network based, or intranet based. Since the system is distributed, the proposed data representations should be efficient, i.e., easy to compress and transfer in a network environment. The other characteristic is "scalable," which means the same data sets and geoprocessing modules can be scaled to serve different users, who may have different bandwidth connections, display devices, or demands. Therefore, different resolution data should be easily produced from the proposed data representations. Wavelet theory provides a rigorous mathematical framework that satisfies the above requirements.

Wavelet analysis is a new mathematical branch dealing with efficiently representing information. Although the earliest work can be dated to the 1910s, the modern wavelet theory was developed in the past 20 years. Now, it has become a useful mathematical tool for various engineering disciplines, such as signal processing, computer graphics, image compression, the finite element analysis, the boundary element analysis, non-destructive detection, video compression, etc. In the 1980s and the early 1990s, Grossmann and Morlet [26], Daubechies [10, 11], Mallat [37], Donoho [17], Meyer [38], and Chui [5] developed wavelet theory for continuous functions and discrete data on regular settings. These wavelets are called first generation wavelets. Since the middle of the 1990s, a group of researchers have begun to construct wavelets for discrete nonuniformly sampled data. Lounsbery [35], Schröder and Sweldens [43], Sweldens [49], Schröder and Zorin [44], and Daubechies et al. [9] have done pioneering work in this area. These wavelets are called second generation wavelets.

Although there are different types of wavelets, the underlying ideas are the same. The essential idea of wavelet theory is to construct a family of good basis functions in order to make data projections on these bases have efficient and hierarchical representations. This fits the data representation requirements for scalable distributed GIS services.

Our research has resulted in contributions to GIS data management and related geoprocessing functions. The proposed Wavelet Image (WImg), Wavelet Digital Elevation Model (WDEM), Wavelet Arc (WArc), and Wavelet Triangulated Irregular Network (WTIN) data representations compose a whole set of multiresolution data

formats for a GIS. This makes scalable online GIS services more practical and efficient. Particularly, the WTIN format provides a more compact representation than the general wavelet-based 3D geometry compression format in computer graphics domain in that it uses one single wavelet coefficient, instead of three, for each geographic point, by using special features of geographic height field data. This makes a very compact data representation possible and also greatly improves the computing efficiency. We also provide algorithms to transfer currently available data sources to these multiresolution formats. Several prototypes have been developed to verify these ideas and the results show that the proposed data representations are implementable and quite acceptable for scalable distributed GIS services.

1.2 Overview

Chapter 2 gives a detailed analysis of why scalable distributed GIS services are useful and what the basic features are; and then a framework based on wavelet theory is proposed. Categorizing and layering techniques are basic tools to organize GIS data. Then based on the categorization of data, we provide a comprehensive literature survey of multiresolution representations of different types of GIS data. Finally, a system architecture is given to explain where the proposed data representations fit in. Object oriented modeling, network-based distributed systems and XML technology for exchanging data are complementary technologies to the whole system.

In Chapter 3, we introduce the first generation wavelet theory. At first, We summarize the traditional wavelet theory, which deals with uniformly-spaced discrete data and continuous functions that satisfy simple scaling and shifting laws. Then we provide a brief discussion on orthogonality, polynomial accuracy, and smoothness of wavelet functions. Finally, several commonly used wavelets are presented.

In Chapter 4, we extend applications on one-dimensional discrete signals to two-dimensional discrete signals. The lifting scheme is introduced to improve computational efficiency of wavelet filter operations. This leads to the discussion of applying first generation wavelets to regular setting (uniformly spaced) data in a GIS. We

present a two-dimensional raster map viewer using first generation wavelets. This is a prototype for scalable distributed GIS services. Here, we are dealing with satellite image data and aerial photo data used in a GIS. A multiresolution format WImg is put forward to make scalable distributed GIS services on this type of data possible and efficient. Detailed discussions of a map mosaic technique, zooming and panning in real time, compression, and comparison of numerical results are also given.

Chapter 5 deals with cell-based Digital Elevation Model (DEM) surface data. Because the USGS (United States Geological Survey) DEM data format uses uniformly spaced data to describe three-dimensional geographic surfaces, first generation wavelets can be applied in processing this type of data. These data are much like the two-dimensional image maps in Chapter 4. However, floating number storage and spatial analyses of geographic surfaces in GIS applications imply that the WImg format is not always suitable for this kind of data. This leads to the development of another data representation, WDEM. Several spatial analyses based on a WDEM are discussed.

In Chapter 6, we introduce the second generation wavelet theory. At first, we examine the lifting scheme and use it to construct traditional wavelet filters. Then we unveil the relationship between wavelet theory and approximation theory. This leads to a discussion of constructing second generation wavelets. Then we investigate the relationship between subdivision schemes and wavelets and point out a simple approach to construct interpolation wavelets.

In Chapter 7, we follow the idea of Chapter 6 and put forward the WArc data representation. This is a direct application of second generation wavelets to curves in a GIS. Curves in a GIS are actually defined by sequences of points, which are stored in vector formats. In some commercial software, the term "arc" is used for this kind of data. The proposed WArc format has multiresolution capability and is easily deployed in scalable distributed GIS services.

In Chapter 8, we put forward a multiresolution triangulated representation—Wavelet Triangulated Irregular Networks (WTINs). In this chapter, we first review the TIN data format and then discuss the characteristics of terrain surfaces. Af-

ter that, we propose the WTIN format based on second generation wavelets. This includes the storage and transmission data format and data structures for online computing. We also present an algorithm to convert publicly available high resolution USGS DEM data to a WTIN. Algorithms for obtaining good initial configurations, constructing the subdivision structure, and compressing data are discussed. Finally, results from numerical experiments are presented.

In Chapter 9, we summarize the contributions of this work. Several conclusions and future research directions are also discussed.

Chapter 2

A Multiresolution Framework

2.1 Scalable Distributed GIS Services

Today, because of the rapid development of the Internet, a large number of computing applications can be delivered as services over a network environment. GIS products follow the same trend. In an open environment, all independent GIS products share resources with others while they are individually managed by different agencies. Then the whole GIS environment becomes a distributed service system. This is an ideal environment for future GIS development.

The network infrastructures of GIS services may be the Internet, wireless communication networks, or intranets. However, transmitting large amounts of data over these infrastructures becomes the biggest obstacle for providing distributed GIS services due to bandwidth limitations. At the same time, different users may have different requirements. Then a distributed GIS service should be adaptable to different users if it is delivered over a network environment. This is a basic requirement for scalable services. There are several kinds of scalability for distributed GIS services. The first one is bandwidth scalability. Different network connections, such as a modem connection, a T1 connection, and a wireless connection, have different transmission speeds and traffic flows; therefore, different resolution data may need to be transferred on different networks. The second kind of scalability is device scalability. Different users use different display devices to view the data. A big screen

may need much detailed data, while a cellular phone screen may only need a coarse level of data. The third kind of scalability is demand scalability. Different users have different interests. One user may want to zoom in after viewing a coarse level of data, while another user may want to change to another scene after viewing the first scene. Therefore, different resolution data will be transferred based on a user's demand. All these kinds of scalability need to be based on a unified data set instead of several data sets that are actually different versions of the same data. This tells us that efficient multiresolution representations are needed for scalable distributed GIS services.

2.2 Categorizing and Layering GIS Data

The first step to manage GIS data is to categorize them. Generally, GIS data can be divided into two types: geographic data and non-geographic data. Non-geographic data are also called attribute data. Attribute data are more like "satellite" data of the geographic data. They can be texts, pictures, numbers, or other descriptive data. Processing them is beyond the scope of this research because wavelet representations are less suitable for attribute data; therefore, we focus only on geographic data. Geographic data are very rich and generally can be categorized into four types: points, curves, surfaces, and volumes. Currently, volumes are seldom used in practice and they are closely related to surfaces, so we will not discuss them. We will not discuss unconnected sets of points either because processing them is trivial. We can simply tag different resolution-labels on points if we want to put them into a multiresolution framework. Curves can be in either a two-dimensional environment or a three-dimensional environment. In a GIS, curves are stored as point sequences, which are connected by different types of basic curves, such as straight lines or B-splines. Therefore, a unified approach to process point sequences needs to be developed to deal with both two-dimensional and three-dimensional cases. For surfaces, satellite images and aerial photographs are the general forms of two-dimensional format; cell-based Digital Elevation Models (DEMs) and Triangulated Irregular Networks (TINs) are general forms of three-dimensional data. Currently, commercial GIS software has

included all of these data formats, and specially designed data structures are used to store geometry and topology information.

After categorizing data, we need to use an important technique—the layering technique—to integrate all types of data. The layering technique is a powerful tool. For a given region, we can process the point features, curvilinear features, and surface features separately and put them into different layers. When we present them, we can put one layer on top of other layers and set different transparency parameters to make them visible altogether. This layering technique is the foundation for our analysis since it makes us comfortable considering only one category of data at a time. Attribute data, such as the population density, can also be plotted on different layers and then combined with the corresponding geographic data.

Through the above discussion, the data types that we need to process are clear. However, storing all these data without consideration of their internal structures is not a good strategy for scalable distributed GIS services. For example, when a user wants to see a detailed map of a location after he/she has previewed the coarse level data, a detailed map is loaded while the previous data are discarded. Nevertheless, both levels of data are correlated. It will waste bandwidth if we throw away the previous data in a distributed GIS service.

Based on the above analysis, we see that the traditional GIS data formats need to be improved to fit into a network environment. A basic strategy is to put GIS data into a multiresolution representation.

2.3 Previous Research on Multiresolution GIS Data Representations

Many researchers have performed studies on multiresolution GIS data representations. Similar studies exist in other related fields as well. The earliest work that we found was in the cartography domain, where the simplification of vector data is an important problem.

2.3.1 Curves

In 1973, Douglas and Peucker [18] designed a method to simplify a curve for cartographic tasks. In this method, a curve is represented by its two ends. The straight line connecting these two ends is the initial simplification. If the distance of the farthest point on the original curve to this line is within a predefined tolerance, then the simplification is accepted; otherwise, the curve will be separated into two segments by this point and the above procedure will be repeated until the tolerance requirement is satisfied on every segment. Some other algorithms exist, but the performance/cost ratios are not as good as that of the Douglas-Peucker's method. Heckbert and Garland [27] provide a good survey on this topic.

Finkelstein and Salesin [22] put forward a multiresolution representation of curves based on wavelets. Their idea is to treat a curve as two separate discrete sequences, x and y, and then apply the spline wavelets of regular grids to both sequences consistently to obtain a multiresolution representation of the original curve. They discussed several operations, such as smoothing a curve, editing a curve in different levels, scan conversion, and curve compression. However, in their work, they restrict their attention to the endpoint-interpolating cubic splines, which is defined on a knot sequence that is uniformly spaced everywhere except at its ends. They also relaxed the continuity constraints for approximating curves because their primary application is scan conversion, which does not require a particular continuity. Relaxing continuity allows a high compression rate for their application.

Buttenfield [4] discussed the progressive transmission of vector data on the Internet. The idea is to use the Douglas-Peucker's method (in [4], this method is referred to as the RDP algorithm) to decompose vector data into stripes in different scales, then use a tree structure and related encoding schemes to store the connectivity information. The proposed structure is helpful for progressive transmission, but the paper did not present an implementation and related analyses.

Bertolotto et al.[3] discussed the progressive transmission of vector data as well. But they focused on the conceptual model and the consistency between different

scales. Some operations such as contraction, thinning, merging, and abstraction were discussed for points, lines, and regions.

2.3.2 Surfaces

Three-dimensional GIS representations have been studied since the middle of the 1980s. Due to the needs of the industrial applications of two-dimensional mapping, several researchers began to study the three-dimensional GIS in the 1980s. Results have been reported in the fields of geology, mining, and civil engineering [41]. Several researchers began to discuss applying computational geometry to GIS research, especially in surface interpolation and spatial adjacency [25]. Most research on three-dimensional multiresolution data representations appeared in the 1990s although some work was undertaken in the 1970s [6] in some GIS-related fields, such as computer graphics.

Representations for Regular Setting Data

In the last decade, along with the rapid development of computer hardware, a lot of work on three-dimensional computer graphics and GIS data representations has been published. For regular setting (uniformly spaced) data, approaches similar to those used in image processing can be used. The two general forms of two-dimensional GIS data, satellite images and aerial photos, can be processed by directly using multiresolution image decomposition. However, real-time interactivity is often required for GIS applications. For example, dynamic zooming and panning are common operations in scalable distributed GIS services, whereas a static image does not require such operations. Therefore, more studies on real-time interactivity need to be performed. In the three-dimensional case, DEM is a data format based on a regular setting. Approaches similar to those used in image processing can be applied here. However, surfaces are generally stored as floating point numbers, whereas images are generally stored as integers. Sometimes, GIS applications require the interpolation feature as well. Therefore, different filters may be required for DEM data.

Serious research on wavelets for regular setting data started in the 1980s. Grossmann and Morlet [26] investigated geophysical signals by replacing the Fourier bases with wavelet bases with promising results. Mathematicians such as Meyer [38] and Daubechies [10] developed the orthonormal wavelet theory. Mallat [37] developed a framework for a multiresolution analysis on regular-setting data. Chui [5] developed the spline wavelet theory. Cohen et al. [7] discovered biorthogonal compact wavelet bases. All of these results constructed the basic foundation of modern wavelet analysis on regular-setting data. For two-dimensional data on a regular setting, we generally use the tensor product of two one-dimensional wavelets to process the data. The tensor product of two one-dimensional wavelet bases is called a separable two-dimensional wavelet basis. Many image compression schemes are based on this type of wavelet and have achieved good results.

Representations for Irregular Setting Data

For data on an irregular setting, pioneering work has appeared since the beginning of the 1990s [12, 15, 16, 13, 23, 28, 31, 42, 50]. Scarlatos and Pavlidis [42] extended Douglas-Peucker's method to the two-dimensional case. They used a recursive triangulation procedure. Starting from a base triangulation, in each step they evaluated the maximum error point based on the current triangulation, and then treated it as a new insertion vertex and re-triangulated all the triangles according to some predefined templates. Abdelguerfi et al. [1] did a comparative analysis of a number of existing methods and showed that a variant of Scarlatos' method exhibited better overall performance than other methods based on their criteria.

De Floriani and Puppo [15] proposed a similar but more general method by considering curve approximation of edges. In [16, 13], De Floriani et al. proposed a general framework for multiresolution hierarchical representation and further discussed the data structures for solving the high storage cost problem of the model. The basic idea is still to use triangular splitting according to several predefined templates. In 2000, De Floriani et al. [14] developed an encoding scheme to compress TINs, particularly for the connectivity information. Park et al. [40] improved the encoding scheme by us-

ing vertex reordering and a general bracketing method based on the assumption that most terrain triangulations are very similar to their Delaunay triangulation. They obtained better compression of connectivity data than previous researchers.

Snoeyink and van Kreveld [45], and Jünger and Snoeyink [31] developed a progressive visualization of TINs based on heuristic rules. Vertices were first selected based on these heuristic rules and then TINs were constructed on these vertices by Delaunay triangulation. A simple incremental algorithm including graph traverse and output permutation was used in the process.

Hoppe [28] developed a method to construct progressive meshes based on two basic operations: edge collapse and vertex split. Later in [30], he proposed a scheme for view-dependent refinement control of progressive meshes. The data representation is lossless since every edge collapse can be reversed by a vertex split using recorded connectivity information. The disadvantage of this approach is that the run time is too long. This is the price of greater precision. Error bound control is also a problem for the intermediate representations of progressive meshes.

Lounsbery [35] began to use wavelet transforms as a framework for surface analyses; these wavelets are the prototype of second generation wavelets. The basic idea is to use iterative quaternary subdivision of a base mesh to construct an approximation of the original surface. This method is faster than Hoppe's, but the cost of resampling the original data into subdivision connectivity is high. It does not resolve creases at arbitrary angles either. The construction of the base mesh and the parameterization are based on the work of Eck et al. [20]. They used Voronoi tiling and harmonic mapping as analytical tools. The computing time for Eck's approach is long.

Sweldens [49] later generalized the traditional wavelet theory to data in an irregular setting, which he called "second generation wavelets." These wavelets are based on the lifting scheme; this work closely relates to Lounsbery's work. Schröder and Sweldens [43] applied second generation wavelets to solve problems on a spherical domain. The concept is to use subdivision to construct a prediction filter and then use the lifting structure to construct wavelet bases. By this approach, the customized design of wavelet bases is possible and the corresponding inverse transforms are easily to

find. They investigated the lazy, linear, quadratic, and Butterfly schemes to construct new wavelet bases. Later, Lee et al. [33] developed a new parameterization algorithm, which was claimed to be better than Eck's remeshing algorithm. Zorin [54, 55] further investigated the Butterfly scheme at the extraordinary vertices (vertices at which the number of intersecting edges is not equal to six) and boundary vertices, and proposed a modified Butterfly scheme. He proved his scheme can achieve C^1 continuity at extraordinary vertices, which cannot be achieved in the original Butterfly scheme in [19].

Bajaj [2] proposed a method where vertices and edges were organized into layers, and then inter-layer and intra-layer connectivity information was used to decompose the data into a multiresolution format. An encoding scheme was also given for compression.

Some other research related to multiresolution data representations can be found in the geometry compression domain as well. One interesting example is in [50], where Taubin and Rossignac put forward the topological surgery method to compress the geometry. A vertex spanning tree was used as an assistant data structure. The connectivity information was compressed without loss of information. Vertex positions were compressed with variable loss of accuracy due to quantization. Combined with other schemes, this method provides good compression effects for multiresolution data models.

2.4 Proposed Solution Framework

From the above brief review, we can see that the last ten years was a very active period of wavelet analysis, especially recently for irregular setting data. On the other hand, scalable distributed GIS services are currently still in their infancy. Some leading GIS commercial software companies, such as Environmental Systems Research Institute (ESRI) and Intergraph, have developed their own online GIS products to meet this new demand; however, the performance is not good enough to provide scalable distributed GIS services yet.

Our wavelet-based multiresolution framework of data representations includes four components: WImg, WDEM, WArc, and WTIN. The WImg format is for satellite images and aerial photos, which are currently stored in the raster format for two-dimensional applications. The WDEM format is for three-dimensional data in a regular setting, which are traditionally stored in the cell-based DEM format. The values in WDEM are generally floating point numbers. The WArc format is for curves. The WTIN format is for nonuniformly spaced data. These data are traditionally stored in TINs. Because both image data and DEM data are regular setting data, they are processed by first generation wavelets. The other two types of data, curves and triangulated three-dimensional surfaces, are generally processed by second generation wavelets due to their irregular settings.

Based on these four multiresolution data representations, we can host scalable distributed GIS services on the Internet. The architecture of the system is a multi-tier system. Figure 2-1 gives a simple view of the service model. In order to achieve platform independence, OpenGIS techniques and XML can be incorporated.

The following chapters will first introduce the traditional wavelet theory and its application to the WImg and the WDEM data representations, then we will introduce the second generation wavelet theory and apply this to design the WArc and the WTIN data representations. Finally we will point out the contribution of this research and compare it with other research.

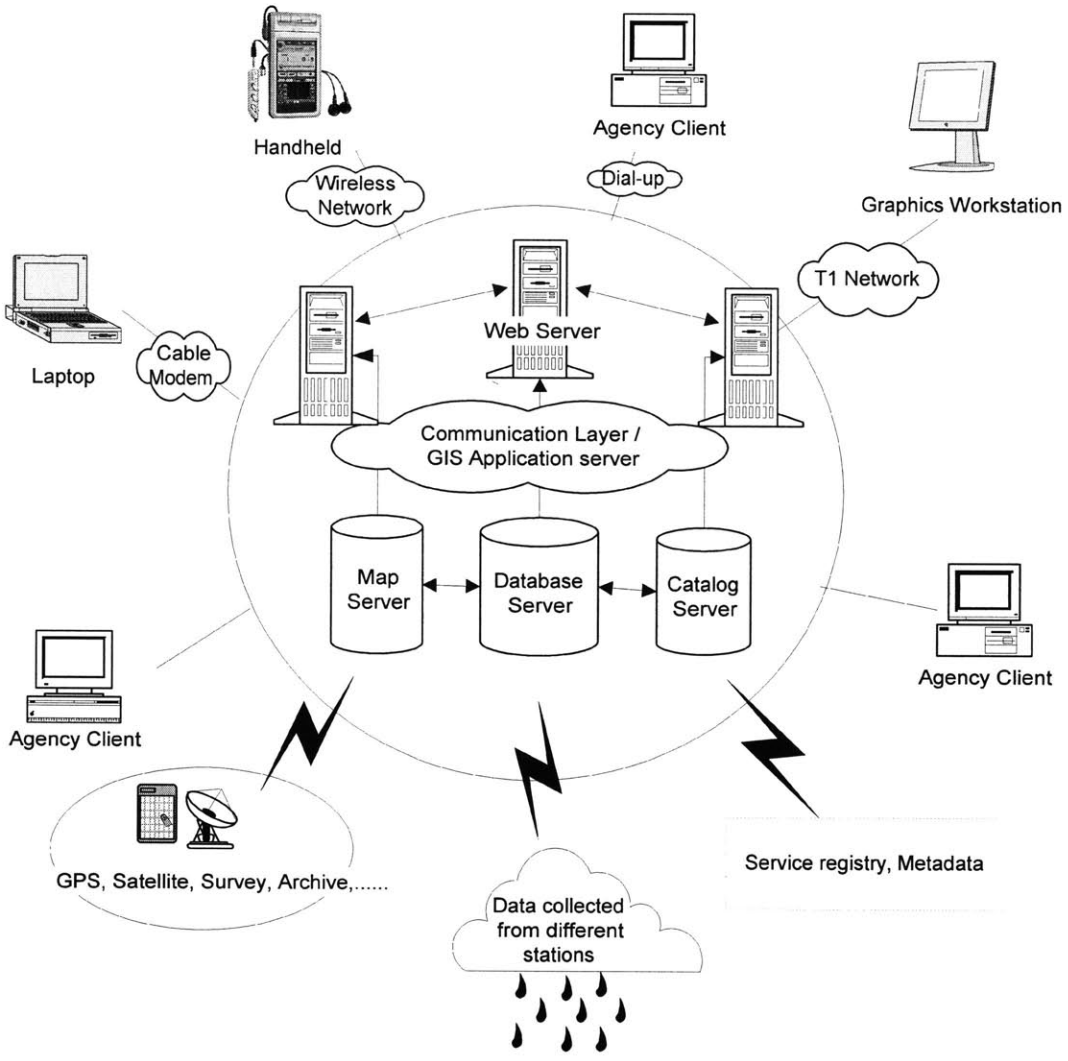


Figure 2-1: A possible system architecture for scalable distributed GIS services

Chapter 3

First Generation Wavelets

The modern wavelet theory has been developed since the 1980s. After two decades of development, it has become a powerful mathematical tool for researchers and practitioners. Its applications are very broad, such as signal processing, computer graphics, image compression, numerical methods for partial differential equations, non-destructive detection, video processing and so on. The basic idea of wavelet theory is to use structured and efficient bases to analyze functions and signals by transforming them into a time (space)-scale (frequency) representation.

Although research on wavelets has been continuing for more than 20 years, most work has been devoted to the regular setting (uniformly spaced) data, which is similar to the case of discrete Fourier transforms. These wavelets are called first generation wavelets. First generation wavelets include the most famous wavelets, such as those used by Grossmann and Morlet (1984 [26]), Daubechies (1988 [10]), Mallat (1989 [37]) and Chui (1992 [5]), which are all designed for the regular setting.

3.1 Basic Theory

As we all know, Fourier analysis represents data as a summation of sinusoid waves. Based on the concept of frequency, a time or space domain function (or a discrete signal in the discrete case) is transformed to a frequency domain function (or signal). However, Fourier analysis cannot tell us when or where a specific frequency

occurs. That is to say, the time/space information is hidden in the frequency representation. In 1946, Dennis Gabor developed the *Short-Time Fourier Transform* (STFT) [11], which can provide both the time and the frequency information simultaneously. However, the STFT has a disadvantage that the size of the windows used in the transform is fixed for all frequencies. This means the precisions for all frequency components are the same. Hence the STFT is not flexible enough for many applications. Wavelet analysis takes one step further to represent information in both the time and frequency domain with variable precisions.

3.1.1 Continuous Wavelet Transform

The *continuous wavelet transform* (CWT) is defined below, following the same style as the definition of the Fourier transform.

$$w_f(s,p) = \int_{-\infty}^{\infty} f(t)\psi_{s,p}^*(t)\,dt\ . \tag{3.1}$$

In this definition, s is the scale parameter; p is the position parameter; $*$ means complex conjugate. In this thesis, we only consider real functions, so $*$ will be dropped for subsequent discussion. The output is a function of the scale and the position $w_f(s,p)$. In the first generation wavelet theory, $\psi_{s,p}(t)$ can be written as:

$$\psi_{s,p}(t) = \frac{1}{\sqrt{|s|}}\psi(\frac{t-p}{s})\ , \tag{3.2}$$

where the function $\psi(t)$ is called a "mother wavelet". It can be seen that all functions $\psi_{s,p}(t)$ can be produced by shifting and scaling this mother wavelet $\psi(t)$. The CWT actually gives the projection coefficients of a function $f(t)$ on all bases $\psi_{s,p}(t)$, which has some internal structures. If $f(t)$ and $\psi(t)$ satisfy certain conditions, $f(t)$ can be recovered from $w_f(s,p)$ [11]. This is the *inverse continuous wavelet transform* (ICWT).

$$f(t) = \frac{1}{C_\psi}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} w_f(s,p)\psi_{s,p}(t)\,\frac{dsdp}{s^2}\ . \tag{3.3}$$

Here, C_ψ is defined as:

$$C_\psi = \int_{-\infty}^{\infty} \frac{|\Psi(\omega)|^2}{|\omega|} d\omega \ < \infty \ , \tag{3.4}$$

where $\Psi(\omega)$ is the Fourier transform of $\psi(t)$. Equation (3.4) is called the *admissibility condition*. From these definitions, it is clear that the CWT provides information for all scales and positions.

Notice that the CWT can be applied to both continuous time functions and discrete time signals (here, "time" can be replaced by "space" as well). The "continuous" in the CWT means that the output result is defined on continuous scales and positions, not on discrete scales and positions. The variables s and p in equation (3.1) are continuous. In practice one often works with discrete time signals. To apply the CWT in this case, we generally need to discretize s and p. We will see below that a particular choice of discretization leads to *discrete wavelet transform* (DWT).

3.1.2 Discrete Wavelet Transform

Although the CWT can provide information on all continuous scales and positions, the computation is very expensive and such detailed data are not always needed for practical use. Most of the time we only need information on discrete scales and positions. This is why the *discrete wavelet transform* (DWT) is more popular in practical applications.

There are many ways to discretize the continuous wavelet transform into the DWT. The most popular way is to use dyadic scales and positions, which means we use

$$s = 2^{-j} \quad \text{and} \quad p = k \cdot 2^{-j}, (j \in \mathcal{Z}, k \in \mathcal{Z}, \mathcal{Z} \text{ is the set of integers}) \ . \tag{3.5}$$

Then the scale s and position p are replaced by j and k respectively. The CWT in equation (3.1) becomes the DWT defined below (since we only consider real functions, we drop the complex conjugate symbol):

$$w_f(j,k) = \int_{-\infty}^{\infty} f(t)\psi_{j,k}(t)\, dt \ , \tag{3.6}$$

where

$$\psi_{j,k}(t) = 2^{\frac{j}{2}}\psi(2^j t - k) \ . \tag{3.7}$$

For a discrete time signal $g[n]$, we can think of it as the expansion coefficients of the projection of a function $f(t)$ on dual scaling bases $\{\tilde{\phi}_{j+1,n}(t)\}$ (the meaning of dual scaling bases will be clear latter). The projection of $f(t)$ is in the space spanned by the corresponding scaling bases $\{\tilde{\phi}_{s+1,n}(t)\}$. This can be written as:

$$\tilde{P}_{j+1}f(t) = \sum_{n=-\infty}^{\infty} g[n]\tilde{\phi}_{j+1,n}(t) \ . \tag{3.8}$$

Plugging this $\tilde{P}_{j+1}f(t)$ into equation (3.6), we can get

$$w_g(j,k) = \sum_{-\infty}^{\infty} g[n]b_{j+1,k}[n] \ , \tag{3.9}$$

where

$$b_{j+1,k}[n] = \int_{-\infty}^{\infty} \tilde{\phi}_{j+1,n}(t)\psi_{j,k}(t)\, dt \ . \tag{3.10}$$

The *inverse discrete wavelet transform* (IDWT) constructs the original data $f(t)$ or $g[n]$ from a wavelet representation.

(3.11)

$$f(t) = \sum_{j,k} w_f(j,k)\tilde{\psi}_{j,k}(t) \ , g[n] = \sum_{j,k} w_g(j,k)\tilde{b}_{j+1,k}[n] \ , \tag{3.12}$$

where

$$\tilde{b}_{j+1,k}[n] = \int_{-\infty}^{\infty} \phi_{j+1,n}(t)\tilde{\psi}_{j,k}(t)\, dt \ . \tag{3.13}$$

Notice that ϕ and ψ are different from the functions $\tilde{\phi}$ and $\tilde{\psi}$ in equations (3.6)$\sim$(3.13). They are dual pairs respectively. The constraints below make all equations consistent.

$$\int_{-\infty}^{\infty} \psi_{l,m}(t)\tilde{\psi}_{j,k}(t) = \delta[l-j]\delta[m-k] \ , \tag{3.14}$$

$$\sum_{j,k} \tilde{b}_{j,k}[m]b_{j,k}[n] = \delta[m-n] \ . \tag{3.15}$$

Equation (3.14) can be further simplified to

$$\int_{\infty}^{\infty} \psi(t-m)\tilde{\psi}(t-k) = \delta[m-k] \ . \tag{3.16}$$

In practical applications, most data are discrete data, and most of the time we also need only transformed data at discrete scales and positions. Therefore, equations (3.9) and (3.10) are most useful in practical applications. We can see that the operation in equation (3.9) is essentially a filter operation, $\tilde{b}_{j+1,k}[n]$ actually defines a filter, which is closely related to the wavelet function $\psi_{j,k}(t)$. From here, we can see why filters have a close relationship with wavelet functions again. Equation (3.7) indicates the relationship between a specific wavelet function $\psi_{j,k}(t)$ and the mother wavelet $\psi(t)$. After this chapter, we will deal with discrete data using the DWT. Sometimes, we use "wavelet transform" to represent the discrete wavelet transform for discrete data when there is no confusion.

3.1.3 Multiresolution Analysis

In the previous two sections, we have discussed the CWT and the DWT on both continuous time functions and discrete time signals. In our applications, the DWT is more important. Therefore, we will only consider discrete scales and positions afterwards.

If $\{\psi_{j,k}(t), j \in \mathcal{Z}, k \in \mathcal{Z}\}$ spans the whole space of finite energy functions $L^2(\mathcal{R})$, we can project any function in $L^2(\mathcal{R})$ on a subspace at the scale j. This subspace is spanned by $\{\psi_{j,k}(t), k \in \mathcal{Z}\}$. We can write

$$L^2(\mathcal{R}) = \bigoplus_{j \in \mathcal{Z}} W_j \ . \tag{3.17}$$

Here, $\oplus$ means a direct sum, which represents the summation of orthogonal subspaces.

$$W_j = \overline{\text{span}\{\psi_{j,k}(t), k \in \mathcal{Z}\}} \ . \tag{3.18}$$

If we directly sum $\{W_m, m = -\infty, \cdots, j-1\}$ and write it as V_j,

$$V_j = \bigoplus_{m=-\infty}^{j-1} W_m, \tag{3.19}$$

then we have nested spaces:

$$0 \subset \cdots \subset V_{-1} \subset V_0 \subset V_1 \subset \cdots \subset L^2(\mathcal{R}) \, . \tag{3.20}$$

This is a multiresolution representation of $L^2(\mathcal{R})$. Notice that the projection of a function on to subspace V_j is an approximation to the original function. This approximation improves as j gets larger. A formal definition of a *multiresolution analysis* (MRA) need five conditions [11]:

1. $\cdots \subset V_{-1} \subset V_0 \subset V_1 \subset \cdots$;
2. $\bigcap_{j \in \mathcal{Z}} V_j = 0 \quad \overline{\bigcup_{j \in \mathcal{Z}} V_j} = L^2(\mathcal{R})$;
3. $f(t) \in V_0 \Leftrightarrow f(2^j t) \in V_j$;
4. $f(t) \in V_0 \Leftrightarrow f(t-n) \in V_0, \forall n \in \mathcal{Z}$;
5. V_0 has a stable basis (Riesz basis) $\{\phi(t-n)\}$.

This definition is for first generation wavelets. For irregular setting data, this definition needs to be extended. We will discuss second generation wavelets in Chapter 6.

Because the complement subspace of V_j in V_{j+1} is the subspace W_j, we can also write equation (3.17) as:

$$L^2(\mathcal{R}) = V_0 \oplus W_0 \oplus W_1 \oplus \cdots \, . \tag{3.21}$$

The constraints on these subspaces are

$$V_{j+1} = V_j \oplus W_j \quad \text{and} \quad V_j \cap W_j = 0 \, . \tag{3.22}$$

Since V_j comes from $\{W_m, m = -\infty, \cdots, j-1\}$, the basis functions for V_j should

be in the subspace spanned by $\{\psi_{m,k}(t), k \in \mathcal{Z}, m = -\infty \cdots j-1\}$. The equations below describe the relationship between the basis for V_j and the basis for W_j, when using equation (3.5) to produce discrete scales and positions. Generally we call equation (3.25) the *refinement equation* and equation (3.26) the *wavelet equation.* $\phi(t)$ is called the *scaling function.*

$$\phi_{j,k}(t) = 2^{\frac{j}{2}}\phi(2^j t - k)\ , \tag{3.23}$$

$$\psi_{j,k}(t) = 2^{\frac{j}{2}}\psi(2^j t - k)\ , \tag{3.24}$$

$$\phi(t) = \sum_k 2l[k]\phi(2t-k)\ , \tag{3.25}$$

$$\psi(t) = \sum_k 2h[k]\phi(2t-k)\ . \tag{3.26}$$

Notice that the last four equations really constrain the bases to be scaling invariant and shifting invariant. The filter coefficients $l[k]$ and $h[k]$ respectively correspond to the unique $\phi(t)$ and $\psi(t)$. In order to obtain an efficient representation, some orthogonality conditions are applied to the basis functions and their duals. This will lead to biorthogonal wavelets, semiorthogonal wavelets and orthogonal wavelets.

3.2 Orthogonality in Wavelets

Based on the MRA framework of wavelet analysis, different orthogonality conditions can be applied to achieve different features of wavelets.

3.2.1 Biorthogonal Wavelets

The DWT and the IDWT are dual processes. In the last section, the DWT is used to decompose information into a MRA framework. The IDWT is used to reconstruct the MRA data to obtain the original format. In the filter bank world, we refer to the DWT as the analysis transform and the IDWT as the synthesis transform. Two embedded subspaces are actually involved in these two processes.

$$0 \subset \cdots \subset V_{-1} \subset V_0 \subset V_1 \subset \cdots \subset L^2(\mathcal{R})\ ,$$

$$0 \subset \cdots \subset \tilde{V}_{-1} \subset \tilde{V}_0 \subset \tilde{V}_1 \subset \cdots \subset L^2(\mathcal{R}) \; .$$

The complement subspaces between the neighboring subspaces are defined as $\{W_j\}$ and $\{\tilde{W}_j\}$. Orthogonality exists between these subspaces as:

$$W_j \perp \tilde{V}_j \quad \text{and} \quad \tilde{W}_j \perp V_j \; , \tag{3.27}$$

where,

$$V_{j+1} = V_j \oplus W_j \quad \text{and} \quad \tilde{V}_{j+1} = \tilde{V}_j \oplus \tilde{W}_j \; . \tag{3.28}$$

All these subspaces are determined by two scaling functions: a primary scaling function, $\phi(t)$, for $\{V_j\}$ and a dual scaling function, $\tilde{\phi}(t)$, for $\{\tilde{V}_j\}$. Equations (3.23)$\sim$(3.26) describe all the bases and their relationships for $\{V_j\}$. Similar equations exist for $\{\tilde{V}_j\}$. Since there are two sets of orthogonality conditions, these wavelets are called biorthogonal wavelets. The orthogonalities of the subspaces can be written as the orthogonalities of the bases.

$$< \psi(t), \tilde{\phi}(t-k) >= 0 \quad \text{and} \quad < \tilde{\psi}(t), \phi(t-k) >= 0 \; , \tag{3.29}$$

where the inner product, $< f(t), g(t) >$, is defined as:

$$< f(t), g(t) >= \int_{-\infty}^{\infty} f(t) \cdot g(t)\, dt \; . \tag{3.30}$$

Here, $\psi(t)$ and $\tilde{\psi}(t)$ are the mother wavelets for the subspaces $\{W_j\}$ and $\{\tilde{W}_j\}$ respectively. The primary and dual relationship also requires

$$< \phi(t), \tilde{\phi}(t-k) >= \delta[k] \quad \text{and} \quad < \psi(t), \tilde{\psi}(t-k) >= \delta[k] \; . \tag{3.31}$$

There are also two refinement equations and two wavelet equations:

$$\phi(t) = \sum_k 2l[k]\phi(2t-k) \qquad \text{and} \qquad \tilde{\phi}(t) = \sum_k 2\tilde{l}[k]\tilde{\phi}(2t-k) \; , \tag{3.32}$$

$$\psi(t) = \sum_k 2h[k]\phi(2t-k) \qquad \text{and} \qquad \tilde{\psi}(t) = \sum_k 2\tilde{h}[k]\tilde{\phi}(2t-k)\ . \tag{3.33}$$

They define two sets of filters ($l[k]$, $h[k]$) and ($\tilde{l}[k]$, $\tilde{h}[k]$). The coefficients in equations (3.10) and (3.13) are really defined by $h[k]$ and $\tilde{h}[k]$ as:

$$b_{j+1,k}[n] = \sqrt{2}h[n-2k] \quad \text{and} \quad \tilde{b}_{j+1,k}[n] = \sqrt{2}\tilde{h}[n-2k]\ .$$

Notice that j does not appear in the above filter indices because of the dyadic relationship in equation (3.5). The biorthogonality relationships on filters are

$$\sum_k h[k]\tilde{l}[k-2m] = 0 \quad \text{and} \quad \sum_k \tilde{h}[k]l[k-2m] = 0\ . \tag{3.34}$$

The primary and dual relationships of filters are

$$2\sum_k l[k]\tilde{l}[k-2m] = \delta[m] \quad \text{and} \quad 2\sum_k h[k]\tilde{h}[k-2m] = \delta[m]\ . \tag{3.35}$$

When biorthogonal filters are applied to a discrete time signal $g[n]$, we treat $g[n]$ as the projection of a function on V_{J+1} and let $c_{J+1}[n] = g[n]$. Then the primary wavelet transform (analysis) becomes:

$$c_j[n] = \sum_m \tilde{l}[m]c_{j+1}[m+2n] \quad \text{and} \quad d_j[n] = \sum_m \tilde{h}[m]c_{j+1}[m+2n]\ , \tag{3.36}$$

where $j = J, J-1, \cdots, 0$. The primary inverse wavelet transform (synthesis) becomes:

$$c_{j+1}[n] = \sum_m c_j[m]l[n-2m] + \sum_m d_j[m]h[n-2m]\ ,\ j = 0, 1, \cdots J. \tag{3.37}$$

Notice that the above formulations are the primary DWT and IDWT for discrete-time signals. The dual DWT and IDWT have similar formulations except that $(\cdot)$ is exchanged with $(\tilde{\cdot})$. Because we generally do not use them, we will not discuss them here.

3.2.2 Semiorthogonal Wavelets

In biorthogonal wavelets, there are two multiresolutions: $\{V_j\}$ and $\{\tilde{V}_j\}$. If these two series of subspaces are identical, then we will have

$$V_j \perp W_j \ , \quad V_{j+1} = V_j \oplus W_j \ .$$

However, we may still need two sets of bases, ϕ_j and $\tilde{\phi}_j$, for the multiresolution $\{V_j\}$ (or $\{\tilde{V}_j\}$) since $\{\phi_j\}$ are generally not orthogonal to each other. So now the wavelets $\{\psi_{j,k}(t)\}$ are perpendicular to the wavelets $\{\psi_{i,m}\}$ if $j \neq i$ because the subspace W_j is orthogonal to V_j which contains all previous W_{j-1}, W_{j-2}, ... Therefore, W_j is perpendicular to all wavelets at all other scales although the wavelets at the same scale are not perpendicular to each other. This can be written as:

$$< \psi_{j,k}(t), \psi_{i,m}(t) >= 0 \ , \quad \text{if} \quad j \neq i \ .$$

We call such wavelets semiorthogonal wavelets. The most important semiorthogonal wavelets are spline wavelets. Splines are piecewise polynomials and satisfy refinement equations. Convolving a spline function with a box function will produce a higher degree spline. These good properties make a spline a good candidate for the scaling function in a wavelet transform. Compactly support B-splines are practically useful splines. Biorthogonal wavelets can be constructed based on B-splines. These wavelets have FIR (Finite Impulse Response) filters. Because the orthogonality between V_j and W_j can be constructed in semiorthogonal wavelets, B-splines are also used to create semiorthogonal wavelets. Generally, the constructed semiorthogonal filters are IIR (Infinite Impulse Response). Chui [5] provides a good introduction to spline wavelets.

3.2.3 Orthogonal Wavelets

Semiorthogonal wavelets have two sets of scaling functions, while these scaling functions spanned the same subspaces $\{V_j\}$. Orthogonal wavelets go further to reduce

the two sets of scaling functions to one unique set. Therefore, in the orthogonal case, the primary and dual spaces are the same and they use the same scaling function, i.e. the scaling functions are the same in the biorthogonal framework. Therefore, the relationships given below exist among basis functions.

$$< \phi_{j,k}(t), \phi_{j,m}(t) > = \delta[k-m] \,, \tag{3.38}$$

$$< \psi_{j,k}(t), \psi_{j,m}(t) > = \delta[k-m] \,, \tag{3.39}$$

$$< \phi_{j,k}(t), \psi_{j,m}(t) > = 0 \,, \tag{3.40}$$

$$< \psi_{j,k}(t), \psi_{i,m}(t) > = \delta[j-i]\delta[k-m] \,. \tag{3.41}$$

Because dyadic scales are always used to construct wavelets, the above equations are equivalent to

$$\begin{aligned} < \phi(t-k), \phi(t-m) > &= \delta[k-m] \,, \\ < \phi(t-k), \psi(t-m) > &= 0 \,, \\ < \psi_{j,k}(t), \psi_{i,m} > &= \delta[j-i]\delta[k-m] \,. \end{aligned}$$

The orthogonalities among basis functions can also be reflected into the relationships of filters. The relationships given below exist among filters.

$$2\sum_k l[k]l[k-2m] = \delta[m] \,, \tag{3.42}$$

$$\sum_k l[k]h[k-2m] = 0 \,, \tag{3.43}$$

$$2\sum_k h[k]h[k-2m] = \delta[m] \,. \tag{3.44}$$

If we choose the filter h to be the alternating flip of the filter l, shown in equation (3.45), then equation (3.43) is automatically satisfied. Then only equation (3.42) is left in the above equations.

$$h[k] = (-1)^k l(N-k) \,, \quad k = 0, 1, 2, \ldots N, \tag{3.45}$$

where the length of the filter is $N+1$. For discrete-time signals, the DWT and the IDWT are similar to those in the biorthogonal case.

DWT:

$$c_j[n] = \sum_m l[m]c_{j+1}[m+2n] \;, \tag{3.46}$$

$$d_j[n] = \sum_m h[m]c_{j+1}[m+2n] \;; \tag{3.47}$$

IDWT:

$$c_{j+1}[n] = \sum_m c_j[m]l[n-2m] + \sum_m d_j[m]h[n-2m] \;. \tag{3.48}$$

3.3 The Mallat Algorithm

The formulas for the DWT and the IDWT in the last section can be used to compute the wavelet coefficients and approximate signals at different scales. In a real application, a filter bank is often used to represent the process. The filters in the refinement equation and the wavelet equation are generally chosen to be lowpass filters and highpass filters respectively. Then the outputs of the DWT are a low frequency approximation (from the lowpass filter) and a high frequency detail (from the highpass filter). The lowpass filter corresponds to the scaling function, and the highpass filter corresponds to the wavelet function. This implementation is called the *Mallat algorithm*. The DWT process is shown in Figure 3-1.

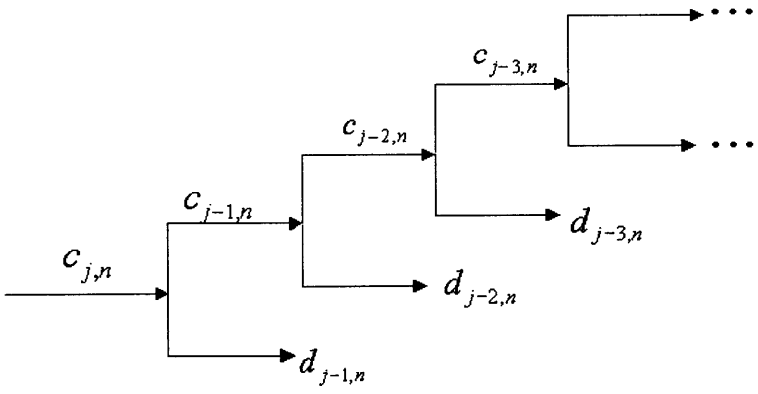


Figure 3-1: The Mallat algorithm

3.4 Accuracy and Smoothness

From the previous introduction for different kinds of first generation wavelets, we can see that wavelets are good bases to represent data in a multiresolution format. However, how efficiently does the transformed data in a wavelet basis represent the original data? This depends on implementation algorithms and the approximation potential of wavelet transforms.

3.4.1 Polynomial Accuracy of Approximation

When we decompose data into a wavelet representation, we want to know how accurately the data can be approximated. Polynomials are the most popular measurement tool for this task. In a MRA, the projection of a function $f(t)$ on the jth level subspace, V_j, is $P_j f(t)$. The approximation error is then given by:

$$\|f(t) - P_j f(t)\| \approx C(\Delta t)^p \|f^{(p)}(t)\| \,. \tag{3.49}$$

The constant C and the exponent p depend on the filters. The exponent p is the degree of the first polynomial function $f(t)$ which has a non-zero error. This is similar to the Taylor approximation of a function. Researchers have found out that p relates to the number of zeros at $z = -1$ in the Z-transform of the lowpass filter ([11]). A pth order lowpass filter $l[n]$ has to satisfy two equivalent conditions:

$$L(z) = \left(\frac{1+z^{-1}}{2}\right)^p Q(z) \,, \tag{3.50}$$

$$\sum_{n=0}^{N} (-1)^n n^j l(n) = 0 \,, \quad j = 0, 1, \ldots, p-1 \,. \tag{3.51}$$

$L(z)$ is the Z-transform of the lowpass filter $l[n]$. $Q(z)$ should not have $(1 + z^{-1})$ in its denominator. Then, the corresponding wavelet function (orthogonal to the scaling function, which is determined by the lowpass filter) has p vanishing moments.

$$\int_{-\infty}^{\infty} t^j \tilde{\psi}(t)\, dt = 0 \,, \quad j = 0, 1, \ldots, p-1 \,.$$

This means that the subspace V_0 spanned by $\phi(t-k)$ contains all polynomials of degree less than p.

3.4.2 Smoothness of Scaling and Wavelet Functions

The smoothness of a function is also called the regularity of the function. This regularity is different from the regularity of data sampling. Data in a regular setting means that the data points are uniformly spaced. The regularity of a function is important in estimating the local properties of a continuous time function and a discrete time signal. Because a wavelet function can be expressed as the linear combination of shifted and scaled versions of the corresponding scaling function, the smoothness of a wavelet function and the smoothness of the corresponding scaling function are consistent. The regularity of a function, s, can be an integer or a fraction. When it is an integer, it has the same meaning as differentiability. When s is a fraction, its meaning is very technical. We can define the smoothness s as the largest s for which

$$\left\|\phi^{(s)}(t)\right\|^2 = \frac{1}{2\pi}\int_{-\infty}^{\infty} |\omega|^{2s}|\Phi(\omega)|^2\, d\omega \quad \text{is finite.}$$

Here, $\Phi(\omega)$ is the Fourier transform of $\phi(t)$. This is really the L_2 sense derivative. The most important factor that determines the smoothness is the structure of the eigenvalues of the matrix $T = (\downarrow 2)2LL^T$, where L is the Toeplitz matrix with the lowpass filter coefficients as its entries. The number of zeros at $z = -1$ for the lowpass filter, i.e. p, which plays an important role in the polynomial accuracy of approximation, contributes eigenvalues $\{1, \frac{1}{2}, \frac{1}{4}, \ldots, (\frac{1}{2})^{2p-1}\}$ in T. This means the number of zeros at $z = -1$ also relates to the smoothness of the scaling function and the wavelet function. The zeros at $z = -1$ improve the smoothness of $\phi(t)$. With p zeros at $z = -1$, $\phi(t)$ cannot have more than $p - \frac{1}{2}$ derivatives in L_2 [47]. Excluding the eigenvalues from the zeros at $z = -1$, the other eigenvalues determines the real value of the smoothness. One simple example is that a box function has a L_2 smoothness $\frac{1}{2}$. Further details about this topic are given in [11] and [47]. The eigenvalues of the matrix T are also important in the convergence and stability analysis of the iterated

wavelet transform.

3.5 Commonly Used Wavelets

This section gives a list of commonly used wavelet filters in engineering applications. We only list the filter coefficients for the analysis process because the corresponding synthesis filter can be obtained from the analysis filter.

1. Haar wavelet: filter coefficients are in table 3.1; the scaling function and the wavelet function are in Figure 3-2.

Index k	l	h
0	$\frac{1}{\sqrt{2}}$	$\frac{1}{\sqrt{2}}$
1	$\frac{1}{\sqrt{2}}$	$-\frac{1}{\sqrt{2}}$

Table 3.1: Filter coefficients for Haar wavelet

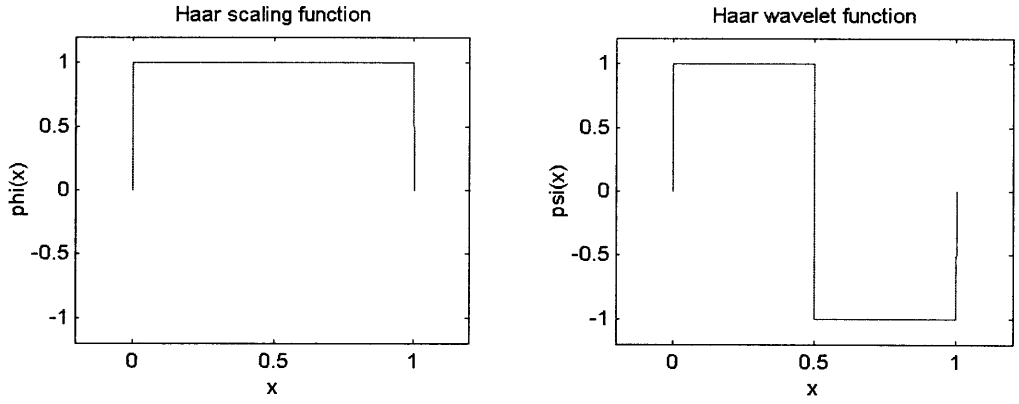


Figure 3-2: Haar wavelet

2. Daubechies wavelets:

   - D_4: 4-tap filter. The wavelet function has order 2 vanishing moments. The filter coefficients are in table 3.2; the scaling function and the wavelet function are in Figure 3-3.

- D_6: 6-tap filter. The wavelet function has order 3 vanishing moments. The filter coefficients are in table 3.3; the scaling function and the wavelet function are in Figure 3-4.

Index k	l	h
0	-0.12940952255092	-0.48296291314469
1	0.22414386804186	0.83651630373747
2	0.83651630373747	-0.22414386804186
3	0.48296291314469	-0.12940952255092

Table 3.2: Filter coefficients for D_4 wavelet

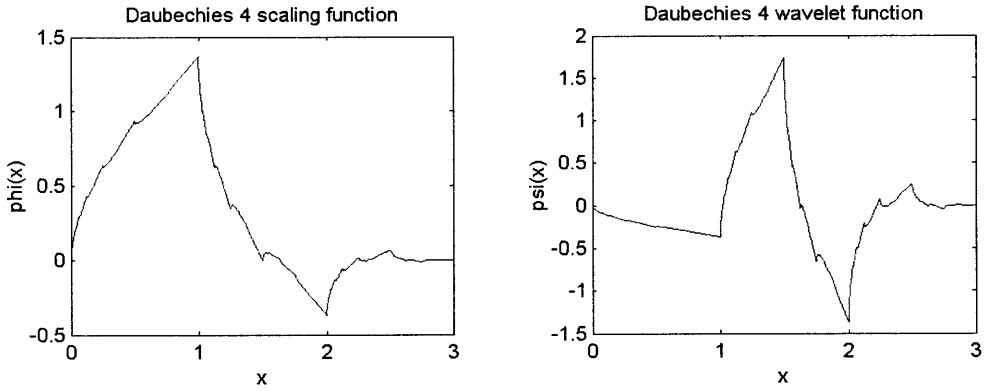


Figure 3-3: Daubechies D_4 wavelet

Index k	l	h
0	0.03522629188210	-0.33267055295096
1	-0.08544127388224	0.80689150931334
2	-0.13501102001039	-0.45987750211933
3	0.45987750211933	-0.13501102001039
4	0.80689150931334	0.08544127388224
5	0.33267055295096	0.03522629188210

Table 3.3: Filter coefficients for D_6 wavelet

3. Biorthogonal wavelets: Bior9/7 filter. The scaling functions and the wavelet functions are in Figure 3-5, the filter coefficients are:

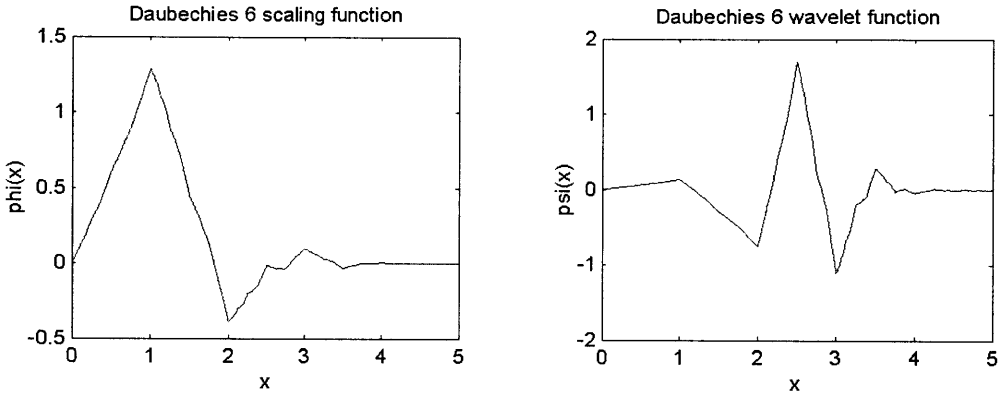


Figure 3-4: Daubechies D_6 wavelet

Index k	l	h
0	0.03782845550726	-0.06453888262870
1	-0.02384946501956	0.04068941760916
2	-0.11062440441844	0.41809227322162
3	0.37740285561283	-0.78848561640558
4	0.85269867900889	0.41809227322162
5	0.37740285561283	0.04068941760916
6	-0.11062440441844	-0.06453888262870
7	-0.02384946501956	
8	0.03782845550726	

Table 3.4: Filter coefficients for biorthogonal 9/7 wavelet

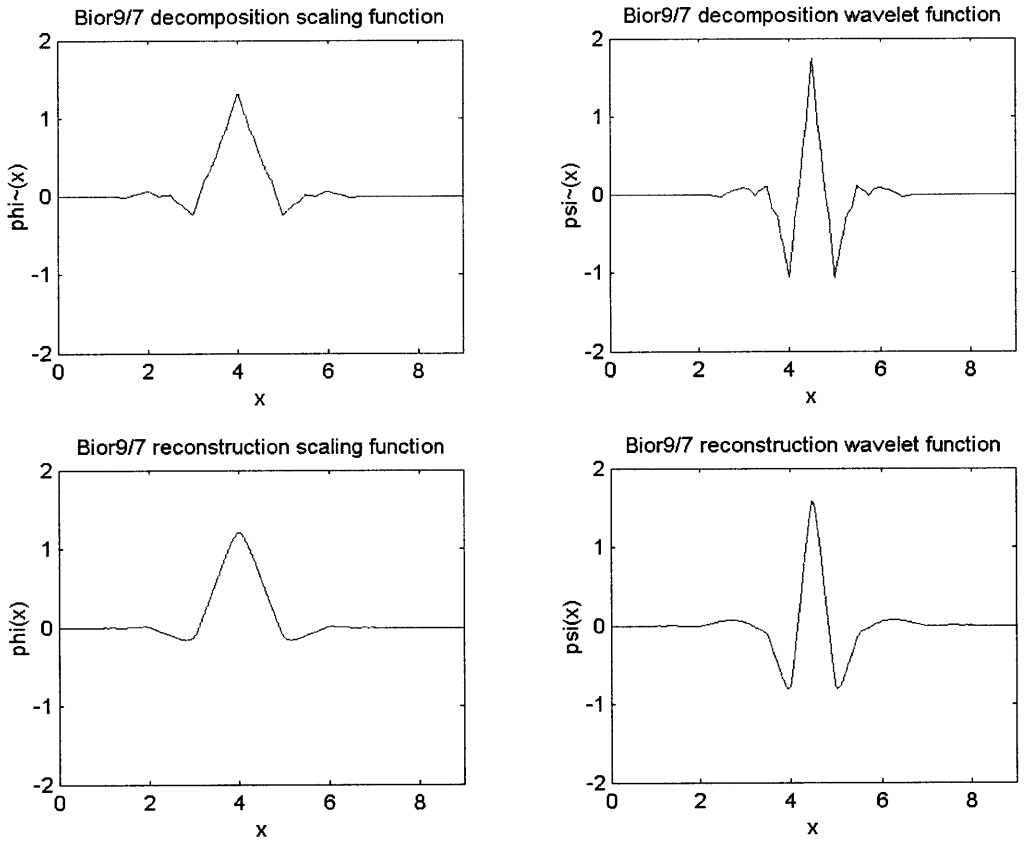


Figure 3-5: Biorthogonal 9/7 wavelet

Chapter 4

Wavelet Image (WImg) Data Representation

Chapter 3 has introduced the first generation wavelet theory which deals with *regular setting* (uniformly spaced) data. In GIS applications, regular setting data include satellite images, aerial photos, and surface data defined on uniform grids. The first two types of data are stored in image formats, in which an integer value is stored for every pixel. The third one, surface data on uniform grids, is generally referred as a Digital Elevation Model (DEM) in the GIS domain, in which floating values are stored for the uniformly spaced points. This chapter is about how to use first generation wavelets to process image maps to achieve an efficient representation and then develop a prototype for dynamic zooming and panning of mosaic image maps. DEM data will be discussed in the next chapter.

Satellite images and aerial photos are 2D images. Generally, the color or grayness of an image map reflects geographic changes. One of the most challenging problems in delivering geographic images over the Internet is how to reduce the transmission time of huge amounts of data over limited-bandwidth computer networks. The first generation wavelet theory provides a good way to address this problem for regular setting data. Using a wavelet decomposition, an image map can be compressed at a high compression ratio, which is generally better than using JPEG format [47]. Another advantage of using wavelets is that a multiresolution representation provides a

way to reuse the previously transmitted data. A high-resolution GIS map has many details that are not useful for a large-scale view. The high-resolution features are useful only when a user wants to zoom into the map for detailed information. Therefore, a wavelet-based multiresolution representation of maps provides a framework to decompose a set of image maps into a *non-redundant* hierarchy of submaps by storing only the differences between various resolutions. This difference information is only transmitted when necessary and will be combined with previously transmitted low-resolution information on the client side to reconstruct a zoom-in view. In this way, unnecessary retransmission of previous data is avoided.

In this chapter, 2D wavelet transforms for images are first introduced. After that, compression schemes and the mosaic map management are introduced in order to develop a prototype interactive image map viewer. Finally, the prototype viewer and related results are presented to show the effectiveness of the proposed approach.

4.1 2D Wavelet Transform on Image Maps

In real GIS applications, data are often two dimensional. In Chapter 3, the 1D wavelet transforms have been introduced. These 1D wavelet transforms can be easily extended to 2D by applying the 1D wavelet transforms on each dimension separately. This is called the *tensor product* of two 1D wavelet transforms. A 2D wavelet transform now decomposes the original space into one average subspace and three detailed subspaces. Discrete data, such as an image map, can be treated as a 2D array. Every row and column of the array is a 1D data set, so the 1D DWT can be firstly applied to all row vectors and then applied to all the column vectors. Figure 4-1 illustrates this process.

In Figure 4-1, the subscripts L and H refer to the lowpass and highpass filtered components respectively. Each of the final four submaps represents a result that is from the processing in two directions. A, H, V, and D denote averages, horizontal details, vertical details, and diagonal details, respectively. This process will be recursively performed on the average block A. In this manner, a multiresolution

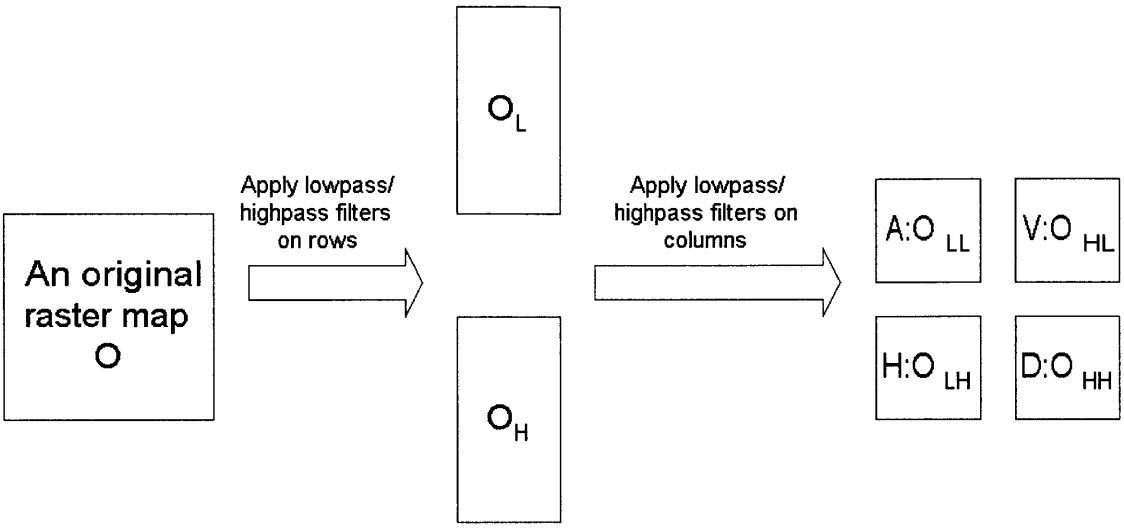


Figure 4-1: Flow diagram of filter bank to decompose an image map

representation of the original map is obtained (Figure 4-2).

In the above discussion, two 1D wavelet transforms are applied on each direction of a 2D data set sequentially. This really leads to a separable 2D wavelet transform. There are also non-separable 2D wavelet transforms [51], which are designed to do the 2D transform jointly in two directions and generally cannot be decomposed to the tensor product of two 1D wavelet transforms. A non-separable 2D wavelet transform could potentially lead to a better result, but they are more complex to design and implement. Hence the vast majority of image processing applications use separable 2D wavelet transforms.

4.2 Lifting Scheme and Integer Transform

To enable online operations such as real-time zooming and panning, a fast computation scheme is needed. Two major factors affect computational speed. One is the filter length. The other is the implementation of the chosen filter. Nowadays, there are many wavelets to choose from. Which wavelet should be used for image maps is an interesting topic in the system design. Essentially, this choice is a tradeoff between compression and speed. From the viewpoint of wavelet compression, the

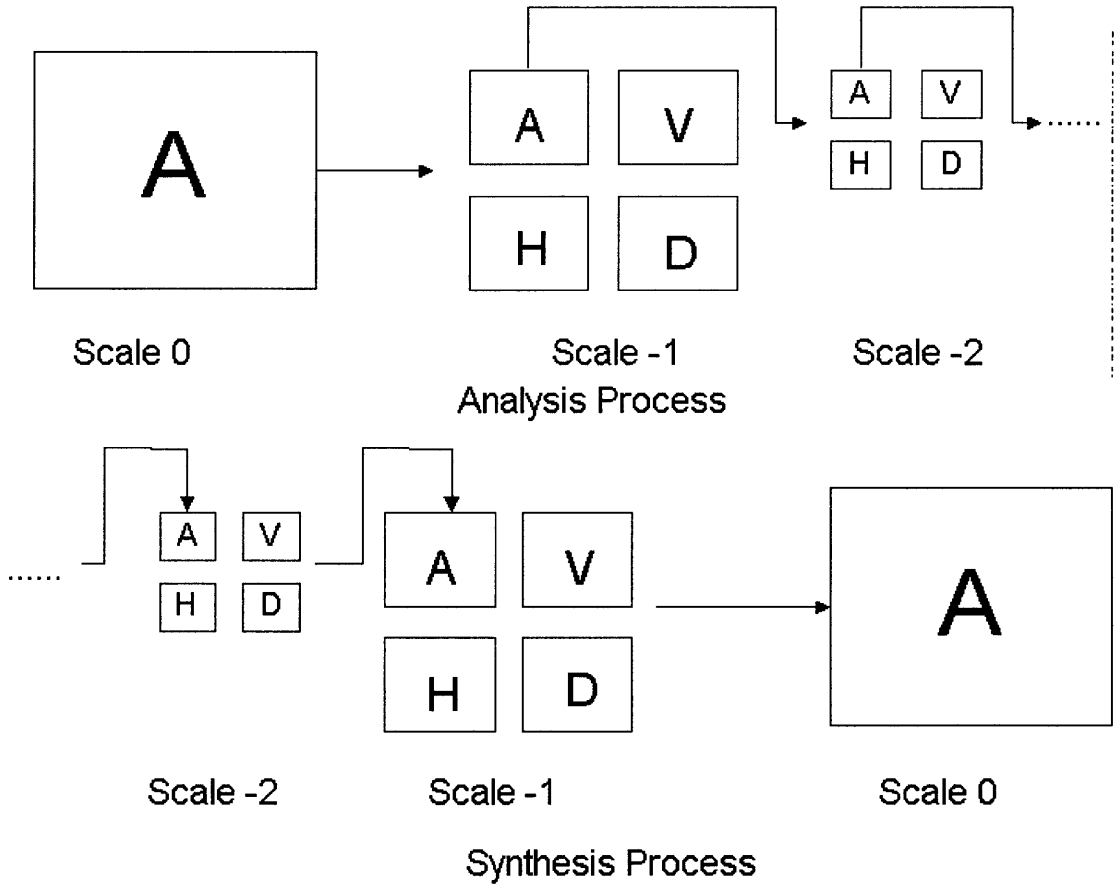


Figure 4-2: Recursive application of the 2D DWT on an image map

Cohen-Daubechies-Feauveau biorthogonal 9/7 wavelet is a good choice for natural images [11]. However, the biorthogonal 9/7 filters (the lowpass filter l_m and the highpass filter h_m) are much longer than the Haar wavelet filters. Therefore, the computation using the biorthogonal 9/7 filters will be much slower than in the case of using the Haar wavelet. On the other hand, although the Haar wavelet does not give an optimal compression ratio, it is simple and fast. Because real-time interaction is desirable, it is reasonable to choose the Haar wavelet filters to implement a GIS prototype. This balances the tradeoff between compression and computational speed and gives satisfactory overall performance from numerical experiments.

The discrete Haar wavelet transform includes only addition, subtraction, and division by two (right shift in binary computing), which makes it a suitable transform for online computing. However, a direct implementation of the Haar DWT needs extra storage space for intermediate results. The storage space is another bottleneck for online computing. Fortunately, a new algorithm, called the lifting scheme, makes in-place computing possible in the discrete wavelet transform. Sweldens [48] has used this scheme to construct new wavelets without using the Fourier transform. Furthermore, all traditional 1D wavelet transforms can be factored into lifting steps [49]. The lifting scheme is also useful for constructing wavelets on nonuniform grids. An additional benefit of this technique is that the wavelet transform can be implemented as an integer-to-integer transform since the original image map is stored as integers.

The basic idea of the lifting scheme is to use a ladder structure to decompose and reconstruct signals. There are three steps in the lifting scheme. The first one is to split (or partition) a signal into different phases. Generally two phases, consisting of even and odd samples, are good enough for 1D signals. Once the signal is separated into even- and odd- indexed signals, the correlation between the two partitions can be explored. The second step in the lifting scheme is a prediction step. Since close correlation exists between the two partitions, we can use one partition to predict the other partition. For example, the even-indexed signal can be used to predict the odd-indexed signal. This introduces a prediction filter in the operation. The difference between the prediction and the original odd-indexed signal gives a highpass signal.

The third step in the lifting scheme is the updating step. In this step, an updating filter is applied on the highpass signal to make the even-indexed signal a better approximation of the whole original signal. The result is a lowpass signal. Figure 4-3 illustrates this process. Another good feature of the lifting scheme is that the reconstruction process can be easily obtained from the decomposition process. We only need to reverse the sign of the updating filter and the predicting filter used in the decomposition process. Perfect reconstruction is automatically achieved through this reconstruction process.

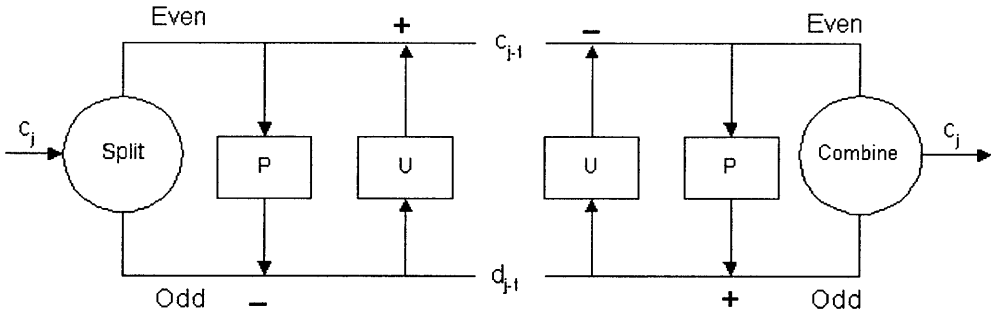


Figure 4-3: Diagram of Lifting Scheme

In Figure 4-3, P is a predictor, U is an updater, and the index j is a resolution index. Equations (4.1)∼(4.4) give the analysis and synthesis formulas. The superscripts e and o represent "even" and "odd" respectively.

$$d_{j-1} = s_j^o - P(s_j^e) \tag{4.1}$$

$$s_{j-1} = s_j^e + U(d_{j-1}) \tag{4.2}$$

$$s_j^e = s_{j-1} - U(d_{j-1}) \tag{4.3}$$

$$s_j^o = d_{j-1} + P(s_j^e) \tag{4.4}$$

For the Haar wavelet transform, the predictor is simply the multiplier 1 and the updater is a right shift (dividing by 2). The equations below give the expressions of the

Haar analysis filter operations in the lifting scheme (Note that the normalization used here is different from that used for the Haar wavelet filters described in section 3.5).

$$d_{j-1,k} = s_{j,2k+1} - s_{j,2k} \tag{4.5}$$

$$s_{j-1,k} = s_{j,2k} + d_{j-1,k}/2 \tag{4.6}$$

For a 2D image map, the two-dimensional formulas are as follows:

For decomposition:

$$d^{j-1}_{m,n} = a^{j}_{2m,2n} - a^{j}_{2m+1,2n} - a^{j}_{2m,2n+1} + a^{j}_{2m+1,2n+1} \tag{4.7}$$

$$v^{j-1}_{m,n} = a^{j}_{2m,2n+1} - a^{j}_{2m,2n} + \left\lfloor \frac{d^{j-1}_{m,n}}{2} \right\rfloor \tag{4.8}$$

$$h^{j-1}_{m,n} = a^{j}_{2m+1,2n} - a^{j}_{2m,2n} + \left\lfloor \frac{d^{j-1}_{m,n}}{2} \right\rfloor \tag{4.9}$$

$$a^{j-1}_{m,n} = a^{j}_{2m,2n} + \left\lfloor \frac{h^{j-1}_{m,n}}{2} \right\rfloor + \left\lfloor \frac{v^{j-1}_{m,n}}{2} \right\rfloor - \left\lfloor \frac{d^{j-1}_{m,n}}{4} \right\rfloor \tag{4.10}$$

For reconstruction:

$$a^{j+1}_{2m,2n} = a^{j}_{m,n} - \left\lfloor \frac{h^{j}_{m,n}}{2} \right\rfloor - \left\lfloor \frac{v^{j}_{m,n}}{2} \right\rfloor + \left\lfloor \frac{d^{j}_{m,n}}{4} \right\rfloor \tag{4.11}$$

$$a^{j+1}_{2m+1,2n} = h^{j}_{m,n} + a^{j+1}_{2m,2n} - \left\lfloor \frac{d^{j}_{m,n}}{2} \right\rfloor \tag{4.12}$$

$$a^{j+1}_{2m,2n+1} = v^{j}_{m,n} + a^{j+1}_{2m,2n} - \left\lfloor \frac{d^{j}_{m,n}}{2} \right\rfloor \tag{4.13}$$

$$a^{j+1}_{2m+1,2n+1} = d^{j}_{m,n} + a^{j+1}_{2m+1,2n} + a^{j+1}_{2m,2n+1} - a^{j+1}_{2m,2n} \tag{4.14}$$

One can see that the reconstruction formulas are easily obtained by reversing the order and the signs of the decomposition formulas. This is true even when the predictor P and the updater U are nonlinear filters such as the above filters with *floor* operations.

Because the original image data are integers and the lifting scheme gives an integer transform, all of the computation can be implemented in the integer arithmetic. For an 8 bit gray scale image, the lowpass data are always in the range that can be represented by one byte; the highpass data are always in the range that can be

represented by two bytes. After quantization, each highpass data can be represented by 1 byte as well. Therefore, this brings a great benefit to implementing compression schemes for the transformed data.

4.3 Compression Schemes

From the previous discussion of 2D wavelet transforms, we know that after a 2D wavelet transform compression techniques need to be applied to the wavelet coefficients to reduce the data size. This is a key step to make an online GIS practically possible. In this research, three techniques are used: wavelet quantization technique, run-length coding, and Huffman coding. This process is shown in Figure 4-4.

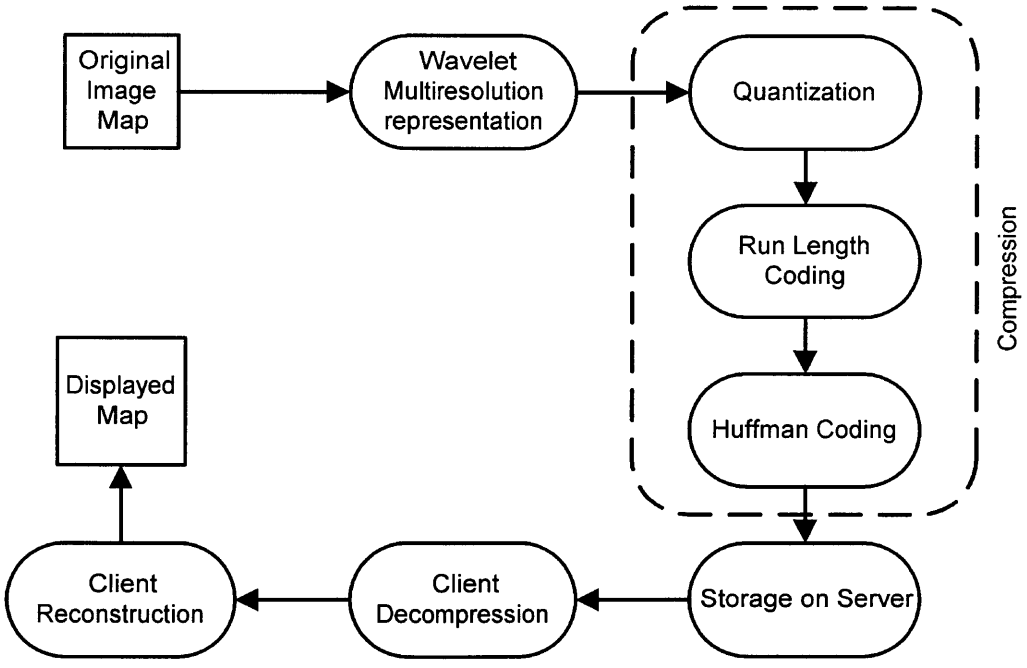


Figure 4-4: Compression of maps

4.3.1 Quantization of Wavelet Coefficients

Quantization is a many-to-one mapping that represents numbers with finite precision by discarding the less significant bits in a number representation. In the wavelet representation, the final lowpass channel data are not quantized because they are the most important and their total size is small. If they are quantized, one will lose valuable information while only achieving a small reduction in size. However, wavelet coefficients (highpass channel data) are sparse and their total size is large. Therefore, wavelet coefficients can be quantized first and then compressed. Since wavelet coefficients are details and less important than the lowpass information at the same level, quantizing them will not distort the image map as much. The quantization technique employed here is known as the uniform quantization [39], which is implemented using a combination of bit shifting and truncation operations. In this paper, the system is designed to use at most seven bits to represent wavelet coefficients and one bit for the run-length identifier (see section 4.3.2). The wavelet coefficients in a range [min, max] are first brought down to 0.0 to 127.0 by a pure linear transformation. Then, quantization will bring them to integers whose precision is 7 bits or less.

The wavelet coefficients in different resolution levels have different importance. For image maps, the finer level to which they belong, the less important they are. Therefore, representing finer-level wavelet coefficients with fewer bits will reduce the data size without significantly affecting the image quality. One important parameter in quantization is the quantization bin size Q. In this system, if $Q = 1$, the quantized data belong to the set $\{0, 1, 2, \ldots, 127\}$, which can be coded in 7 bits. If $Q = 2$, the quantized data belong to the set $\{0, 2, 4, \ldots, 126\}$. The numbers are always even, which means they can be coded in 6 bits. As a rule of thumb, neighboring level wavelet coefficients are coded with a 1 bit difference in precision, so the quantization bin sizes for neighboring levels differ by a factor of 2. We use the notation Q_0 to denote the quantization bin size for the coarsest level wavelet coefficients; Q_1 is used as the quantization bin size for the second coarsest level wavelet coefficients; and this convention continues on other levels. Because truncation is used to convert floating

point data to integers, quantization is essentially a lossy technique. In Figure 4-5, an example of a three level decomposition is given. Then one could use 8 bits/pixel for block 1 (no quantization at all for the coarsest lowpass data); 7 bits/pixel for blocks 2, 3, and 4 ($Q_0 = 1$); 6 bits/pixel for 5, 6 and 7 ($Q_1 = 2$); and 5 bits/pixel for blocks 8, 9, and 10 ($Q_2 = 4$). How to determine Q_0 depends on the desired degree of compression. This is discussed in section 4.6.2. Once Q_0 is chosen, other Q_i can be derived from Q_0 as $Q_i = 2^i Q_0$ ($i = 1, 2, 3, \ldots$).

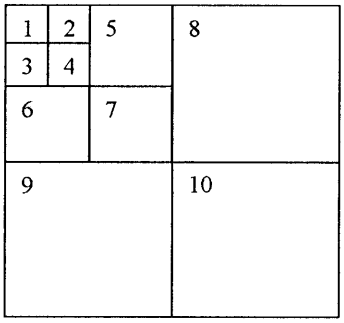


Figure 4-5: Wavelet decomposition (three levels) of map

4.3.2 Run-Length Coding

Run-length coding is a lossless compression technique that can eliminate the obvious redundancy that occurs when data repeat themselves. Instead of storing multiple copies of a repeated value, one stores only a single copy of the value, along with a special number to identify the number of times the value is repeated. For example, if there are 100 consecutive zeros, then it is much more efficient to store the numbers zero and 100 than to store zero 100 times. In this system, as stated above, each wavelet coefficient is coded with seven bits or less, then the extra one bit in a byte representation can be used to identify the run length. In the signed integer system, the extra bit is the sign bit. A minus sign is to used to flag the data as a run length, in which case the absolute value is the real run-length. In our implementation, run-length coding is only applied to zero wavelet coefficients since these occur most

Data	Probability	Huffman Code
0	0.06	010
16	0.1	011
32	0.24	00
64	0.3	11
96	0.16	101
127	0.14	100

Table 4.1: Probabilities of occurrence and corresponding Huffman codes

frequently.

Generally, there is a limit for the run-length. If one uses single bytes to represent the run-length encoded data, then there are only seven bits for the real run length, since one bit is required for the run-length flag. When the run length is larger than 128, another run-length should be used for the representation until all the runs are included.

4.3.3 Huffman Coding

Huffman coding is an elegant algorithm for entropy coding that is based on the statistics of the data, giving fewer bits to the most frequently appearing data and vice versa. For example, Table 4.1 shows the probabilities of occurrences for six integers that appear in an image map representation. Figure 4-6 shows how the Huffman tree is constructed. The idea is to recursively combine the two least-frequent characters as a new node until only a single root node with probability 1.0 remains. One can obtain the code for each character by assigning 0 to each left branch and 1 to each right branch and then tracing the path to the character from the root node. The final code for each character is shown on the last column of Table 4.1. From that column, one can see that the most frequently used characters have the shortest codes. For a data stream {16, 64, 32, 127, 0, 32, 64, 96} (64 bits for 8 bits/data), the Huffman encoded data have only 20 bits, 01111001000100011101. To decode characters from this bit stream, one again traces a path from the root node until a leaf node is reached. After retrieving the data corresponding to the leaf node, one resets to the root node and repeats this process.

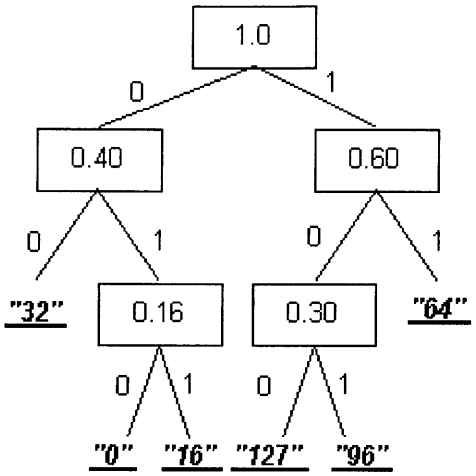


Figure 4-6: Huffman tree

All three of the above compression schemes are widely used in data compression. They are also the general techniques used by the JPEG standard [24], although the quantization bin sizes are chosen differently. However, there are other alternatives. For example, Lempel-Ziv (LZ) is widely used in the Unix compress utility. The Java zip package, which uses a variant of LZ compression called gzip (including Huffman coding), was also used to compress the wavelet coefficients for purposes of comparison. The results show that run-length coding plus Huffman coding can be a little better than the gzip compression for the tested image maps (see the results in section 4.6.2. Arithmetic coding is an alternate algorithm that is specified as an alternatives to Huffman coding in the JPEG standard [24]. However, it is much slower than Huffman coding, so it is not suitable for an online GIS viewer. Besides this, some performance-improved implementations of arithmetic coding are subject to patent restrictions. Based on these considerations, run-length coding and Huffman compression are implemented in the system.

4.4 WImg and Progressive Delivery

Since a wavelet transform can decompose an image map into a multiresolution representation, the WImg (shorthand for "wavelet image") format is defined to store image maps. There are three kinds of data need to be stored. The first one is the control in-

formation: filter type, compression parameters, and resolution numbers. The second one is the coarsest-level map data. The third one is the set of wavelet coefficients for every resolution. Notice that there are three matrices for each resolution: horizontal details, vertical details, and the diagonal details. The data structure used for a WImg is an array.

From the above discussion, it is clear that wavelet analysis provides a good approach to decompose an image map into different resolutions. This fits well in the scalable distributed GIS service model. Even in today's era of broadband communications, users who have the greatest need for GIS data (for example, mobile users) often do not have a high-bandwidth connection to the Internet. Therefore, in order to reduce the traffic on the Internet, a GIS service provider should send only the requested portion of a map to a user instead of the entire map. When the user wants to see more detail on a particular region, the service provider only needs to send the extra "wavelet" information to the user, which will be synthesized with the previous information to get a detailed view of that area. This permits progressive transmission of GIS data in the scalable distributed GIS model.

In the scalable distributed GIS service model, GIS servers and clients may reside at different geographical locations. This architecture is similar to the current Web model. A GIS data server will host GIS data (for example, image maps), while the client can be an applet loaded in a Web browser. In between, there is a middle tier—a service server—to receive users' requests, convey them to a GIS data server, get the result data, and send them to the client. Based on this three-tier architecture, an image service GIS prototype has been built to demonstrate the potential of wavelet theory for GIS applications. There are three main components in the prototype. The first component is a preprocessing program, which decomposes an image map into a multiresolution format, compresses the detailed information, and stores it in a specified structure. The second component is a client program, which is a Java applet that sends out requests based on a user's mouse inputs and reconstructs the map based on the retrieved data. The third component is a server program, which receives requests for maps and sends the requested data to the client. The computational

process is illustrated in Figure 4-7.

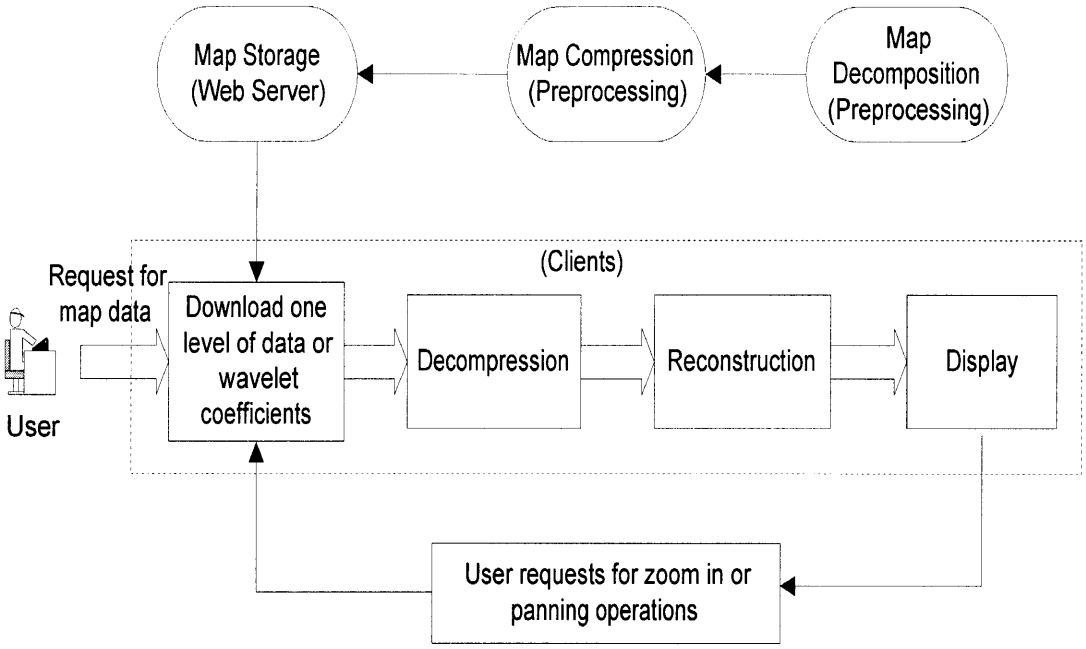


Figure 4-7: Computational process and software structure

In Figure 4-7, an image map is processed by wavelet analysis and then compression schemes are applied to reduce the size of the map. When the client gets the compressed data, it will decompress them and reconstruct the map. All of these processes are based on a thick-client assumption. Currently the bottleneck is the bandwidth of the network, not the client's computing power; therefore, this is practical from an implementation standpoint.

4.5 Map Data Management

The most difficult problem with a GIS is to manage huge data sets. As previously stated, for a large area, it is not wise to put all the data into one large map because the transmission of data needs a tremendous amount of time. Here, we use two techniques to manage map data: layering and tiling.

In Chapter 2, the layering technique was discussed. The basic idea is to separate different types of data into different layers. A real GIS map includes image data,

vector data, point data, and related attribute data. All the data will be put into a spatial database. The layering technique separates different types of data into different layers. Then, image data will not be mixed with vector data because they will be processed separately. This chapter is focused on image data. Vector data and other data can be added on top of the image data layer. This is demonstrated in the last section of this chapter.

Large image maps are often too large to be stored as a single unit. Therefore, the tiling technique is used to divide a large image map into small submaps. We assume that all the image maps are rectangles. Then we can assign a sequence number to each tile. Figure 4-8 shows an example of a big map with 9 tiles. We number them as tile 0, tile 1, ..., tile 8. They are organized as 3×3 blocks. Each tile (also called a "block") is one part of the larger map. When a user selects an area that overlaps with different tiles, the viewer manages the bookkeeping of the submaps so that the processing across block boundaries is seamless. A key problem is to find the submap index for a given point. This can be calculated from equations (4.15)~(4.17). Firstly, the row index and column index are calculated based on the input point coordinates, and then the submap index is derived according to the tile organization.

$$i_r = \left\lfloor \frac{y - y_o}{y_{unit}} \right\rfloor \tag{4.15}$$

$$i_c = \left\lfloor \frac{x - x_o}{x_{unit}} \right\rfloor \tag{4.16}$$

$$i_t = i_r * x_{dim} + i_c \tag{4.17}$$

In the above equations, i_r denotes the row index, i_c denotes the column index, and i_t denotes the tile index. For example, tile 4 in Figure 4-8 has row index 1 (starting from 0) and column index 1 (starting from 0). (x_o, y_o) are the coordinates of the origin; (x, y) are the coordinates of the point of interest; x_{unit} and y_{unit} are the tile sizes in the x and y directions, respectively; x_{dim} is the number of tiles in the x direction.

The key point for bookkeeping is to use the base coordinates of a tile corner point to identify the indices of submaps. For example, the solid-line blocks in Figure 4-9 are

four submaps, and the dashed line area ABCD is the selected area. Because the point P is inside ABCD, the four portions AP, BP, CP, and DP can be abstracted from the four submaps (the four solid-line blocks) according to their map indices. Generally, for each corner point, there are four connected blocks, such as point P. However, for some corner points, such as points O and M, there are fewer than four connected blocks. In this case, one can set the corresponding empty blocks to be null. After having determined the indices of submaps, the viewer can calculate the size of the selected portions of the submaps according to the coordinates of the selected area, and then reconstruct these portions seamlessly.

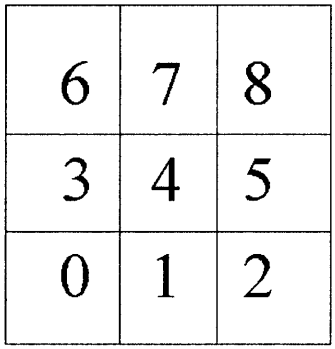


Figure 4-8: Tile Arrangement

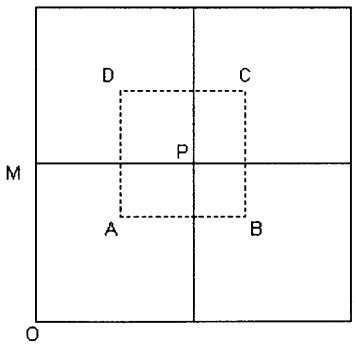


Figure 4-9: Bookkeeping of Maps

4.6 Prototype and Analysis

Based on the above system design, a prototype of a scalable distributed GIS is implemented. The visible part is the graphical user interface (GUI) of the viewer, shown in Figure 4-10. Using the tiling technique, an aerial photographic map of the MIT campus is divided into nine blocks and recorded by indices. Users can choose four different tiles and mosaic them together. Two key functionalities are implemented; panning and zooming. A user can smoothly pan around the map and zoom in or zoom out of the map in *real time*. This viewer is different from a typical map viewer in that it downloads each piece of data only once for all these requests. In a typical map viewer, when a user pans around or zooms in or out, the viewer downloads a new JPEG image even when the user sees the same area. Figure 4-10 (a) is a view of one portion of the map; Figure 4-10 (b) is a snapshot after zoom-in. The maps used here are from the MIT Digital Orthophoto Web site, ⟨http://ortho.mit.edu⟩, by courtesy of MassGIS and MIT.

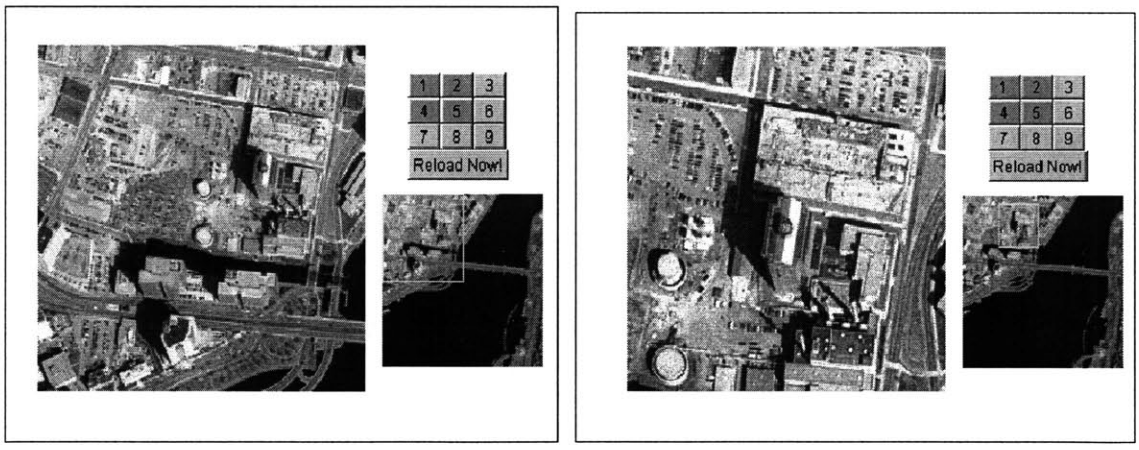


Figure 4-10: (a) GUI of prototype scalable GIS viewer (b) Zoom-in view of the map

4.6.1 Network Programming

Based on the above system design, a client program and a server program have been written in the Java language in order to be platform independent. The client program connects to the server program and sends a map request to initiate the progressive

retrieval of the multiresolution image data. After the connection is made, the client downloads the coarsest resolution data for the map. At the same time, it spawns a new thread to receive the next level of detailed data in preparation for a possible zoom-in request. When a user makes a zoom-in request, the client program first checks if the necessary data has been fully downloaded. If so, after decompression, it uses the wavelet filters to reconstruct a new zoom-in map by using the received detailed data and the previous resolution data. This is the synthesis process expressed in equations (4.11)$\sim$(4.14). If not, the client waits until the necessary data have arrived. When a user makes a panning request, the client program uses the pointer position and the resolution level to determine the visualization range and then reconstructs the newly selected area to provide an updated map view.

The server program is a multithreaded program. The server always listens for requests from clients. Upon receiving a request from a client, the server launches a separate thread to handle it and continues to listen for the next request. The thread for processing a client request loads the map file from the database server and sends it to the client. The basic classes for client/server programming are provided by the Java servlets and network packages. The network programming is based on the TCP/IP protocol.

4.6.2 Performance Analysis

Two compression schemes for compressing the quantized wavelet coefficients are implemented: gzip compression (abbreviated as gzip), and a scheme combining run-length coding and Huffman compression (abbreviated as RH). As stated in the section 4.3, the quantization bin size, Q, is a key factor for compression. The objective is to achieve a good balance between the image quality and the degree of compression. However, how to choose the level of quantization is heuristic, since this depends on the features in the image map. Three possible quantization schemes ($Q_0 = 1, Q_0 = 2$, and $Q_0 = 4$) are compared in Table 4.2.

The choice $Q_0 = 1$ means that a four-level decomposition has quantization bin sizes 1, 2, 4, and 8 from the coarsest level to the finest level. The other two schemes

can be similarly derived. The results show only the data for the submaps 1, 2, 4, and 5 that are chosen in Figure 4-10. From the results, one can see that the peak-signal-to-noise ratio (PSNR) values are the same for both gzip and RH compressions with the same Q_0 value. This is because they use the same uniform quantization technique. However, the RH compression achieves a higher compression rate. This demonstrates that run-length coding and Huffman compression can give better results than gzip compression for these four submaps. Roughly speaking, PSNR values on the order of 30 dB are good enough for the human visual system. If significantly higher than this, humans cannot identify the difference easily. If significantly lower than this, explicit degradation is easily noticeable. For $Q_0 = 2$, all the PSNR values for the submaps are within the range 29 to 32 dB; therefore, in the final system, $Q_0 = 2$ is recommended. The average compression ratio for these four submaps is 5.71:1, which means the resulting data size is only 17.5% of the original data size.

In order to evaluate the performance of the wavelet compression scheme in comparison to JPEG, which is the common format for natural image compression, the GIMP software ⟨http://www.gimp.org⟩ was used to convert the original maps to JPEG. To obtain a fair comparison, the JPEG compression parameter is tuned so that the final JPEG data size is as close to the corresponding RH data size as possible. The results are shown in Table 4.3. This shows that RH compression for the Haar wavelet gives a similar compression quality (PSNR value) to the JPEG format when their compression ratios are similar. Notice that the average PSNR of RH compression with the Haar wavelet decomposition is 0.39% lower than JPEG (around 0.115 lower in decibels), but the average compression ratio of RH is 1.33% higher than JPEG. For the remaining five maps involved in the prototype, similar results are achieved.

In reality, JPEG uses a compression scheme similar to RH; the difference between our approach and JPEG is in the use of the wavelet decomposition versus the DCT (Discrete Cosine Transform) decomposition. The results imply that by using the simple Haar wavelet, a compression performance similar to JPEG can be achieved. However, wavelets provide the extra multiresolution capability that makes the panning and zooming much smoother and faster (because these operations are closely coupled

with the compression algorithms to avoid redundant data transmission). Therefore, for scalable distributed GIS services, the proposed approach has more advantages. Using higher-order wavelets, the compression performance will be greatly improved, but that involves more computation, which is a disadvantage for online computation. Note that the results for the gzip and RH cases shown in Table 4.3 include all the data in the hierarchical wavelet decomposition. Lower-resolution views of the maps require only a fraction of these data.

		Map 1			Map 2			Map 4			Map 5		
		Q_0=1	Q_0=2	Q_0=4	Q_0=1	Q_0=2	Q_0=4	Q_0=1	Q_0=2	Q_0=4	Q_0=1	Q_0=2	Q_0=4
gzip	PSNR (dB)	34.80	29.29	24.72	33.55	29.48	26.11	33.37	29.45	25.91	34.29	30.52	28.14
	Compression Rate (%)	59.84	72.03	84.37	66.75	79.81	90.79	66.60	79.57	90.30	71.04	84.19	94.56
	Compression Ratio (*:1)	2.49	3.58	6.40	3.01	4.95	10.85	2.99	4.89	10.31	3.45	6.32	18.39
	Bits/pixel	3.21	2.24	1.25	2.66	1.61	0.74	2.67	1.63	0.78	2.32	1.27	0.44
run-length and Huffman	PSNR (dB)	34.80	29.29	24.72	33.55	29.48	26.11	33.37	29.45	25.91	34.29	30.52	28.14
	Compression Rate (%)	60.99	74.95	86.77	69.81	82.95	92.40	69.04	82.32	91.88	73.30	86.33	95.14
	Compression Ratio (*:1)	2.56	3.99	7.56	3.31	5.86	13.16	3.23	5.65	12.31	3.75	7.32	20.59
	Bits/pixel	3.12	2.00	1.06	2.42	1.36	0.61	2.48	1.41	0.65	2.14	1.09	0.39

Table 4.2: Comparison of different Q values (Q_0 is the bin size for the coarsest level of wavelet coefficients)

	Map 1			Map 2			Map 4			Map 5		
	JPEG	gzip	RH	JPEG	gzip	RH	JPEG	gzip	RH	JPEG	gzip	RH
PSNR(dB)	29.25	29.29	29.29	29.61	29.48	29.48	29.63	29.45	29.45	30.71	30.52	30.52
Compression Rate (%)	74.55	72.03	74.95	82.75	79.81	82.95	82.21	79.57	82.32	86.06	84.19	86.33
Compression Ratio (*: 1)	3.93	3.58	3.99	5.80	4.95	5.86	5.62	4.89	5.65	7.17	6.32	7.32
Bits/pixel	2.04	2.24	2.00	1.38	1.61	1.36	1.42	1.63	1.41	1.12	1.27	1.09

Table 4.3: Comparison of different compression schemes (for gzip and RH, Q_0=2)

How about computational speed? As stated above, the proposed approach produces smaller data sets than JPEG for lower-resolution views, while the highest-resolution view will be at least as good as JPEG data. So the transmission time is substantially reduced during normal browsing. Besides this, because the proposed approach uses an integer wavelet decomposition with 2 tap filters (compared to an 8 point discrete cosine transform for JPEG) and the remaining compression steps are similar to JPEG, the decompression time for the proposed approach is smaller than the decompression time for JPEG. Therefore the overall performance is substantially better than the approach to transmit separate JPEG images. In practice, the prototype is able to easily achieve a 6:1 compression ratio in high-resolution (0.5 m/pixel) digital orthophotos with little or no noticeable loss of image quality.

A more mature graphical user interface based on the technology described in this chapter has been developed by the high performance computing center at NECTEC (National Electronics and Communications Technology Center) in Thailand. The enhanced GUI is shown in Figure 4-11. A key feature of this version is the ability to download and display vector layers, such as road networks, rivers, and provincial boundaries, which can be seamlessly zoomed and panned along with the base map. These vector data are stored in a remotely accessible Postgres database.

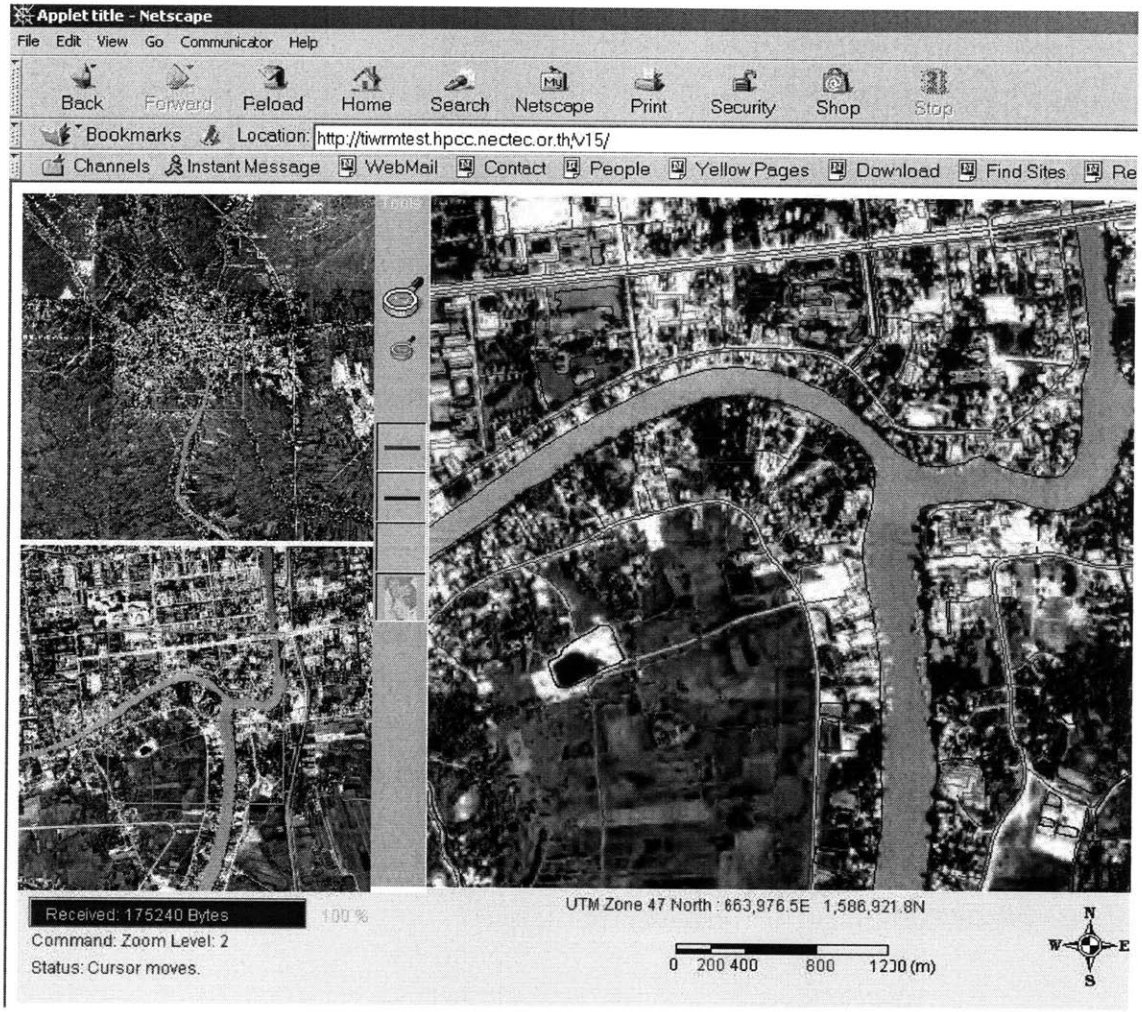


Figure 4-11: GUI for a more realistic scalable GIS viewer

Chapter 5

Wavelet Digital Elevation Model (WDEM) Data Representation

Chapter 4 presents the multiresolution data representation for GIS image data. This is based on wavelet transforms on uniformly spaced data. Another data format defined on uniformly spaced data in the GIS domain is the Digital Elevation Model (DEM) format, which defines geographic surfaces on uniform grids. Therefore, an approach similar to that used in the previous chapter can be used to process DEM data to obtain a multiresolution representation. In the GIS domain, there are two popular formats for surface data. One is the DEM format, the other is the Triangulated Irregular Network (TIN) format. Both have broad applications. DEMs are defined on uniform grids; however, TINs are defined on nonuniformly spaced vertices. Therefore, approaches to process them into multiresolution formats are quite different. In this chapter, we extend the application of first generation wavelets on images to DEMs. For TINs, we need to use second generation wavelets to construct a multiresolution format.

Although images and DEMs are both defined on uniform grids, there are some differences between them. The major differences are:

1. In image maps, the values are stored as integers, which generally ranges from 0 to 255 in gray images; however, in DEM data, values are generally floating point

numbers and the range varies from position to position. Therefore, the integer transform used in processing images cannot be directly used in processing DEM data.

2. In a multiresolution setting, some GIS applications require the interpolation feature, which means that a point in a coarser resolution representation needs to stay at the same position in a finer resolution. This requirement does not exist in the image data representation. Therefore, processing DEM data may need interpolation filters to fit this design requirement. When interpolation is not required, approximation filters, such as those used in WImg, can be used for DEMs as well.

3. A large amount of spatial analysis in the GIS domain is based on surface data; however, little spatial analysis is based on image data, which are mostly used for visualization. Thus, the benefits to spatial analysis from a multiresolution format deserve to be explored in this chapter.

5.1 Digital Elevation Models

First we will look at what is defined in DEM data. Different versions of DEMs exist. We use the United States Geological Survey (USGS) DEM definition here. The content of the USGS DEM is:

number of rows	:	$nrows$
number of cols	:	$ncols$
coordinates for bottom left corner	:	(x_0,y_0)
cell size	:	c
null data value	:	NDV
height values	:	$z_{[nrows][ncols]}$

Table 5.1: United States Geological Survey (USGS) DEM

The core part of a DEM is an array, which defines a rectangular area. Four important issues in this format deserve our attention. The first one is the order of the two dimensional array in a DEM. The element $z[0][0]$ (Java programming style)

is the height value corresponding to the bottom left point of a rectangular area. Thus the first row of a DEM is the bottom line of the area. This is a key point to make DEM data consistent with other data during the integration step. The second one is that a DEM has much more data than an image with the same resolution. This is because a DEM generally includes a two-dimensional array of 32 bit floating point numbers, while a gray image map generally has only 8 bit integers. The third one is that the cell sizes for the x and y directions are the same, which means that a DEM uses an equal interval to sample a height field in both the x and y directions. From here, it is clear that DEM data have redundancy since the same sampling rate is used regardless of whether an area is flat or mountainous. This also leads us to use wavelets to compress DEM data. The last issue is that there are possible "null data values" in a DEM. This means that measurements are not available at those positions. Generally extrapolation or interpolation approaches are needed to fill those values in order to create a reasonable surface.

The basic framework for processing DEM data is similar to that for processing image data. Two wavelet transforms are separately applied to data in the x and y directions. Because there is no essential difference for the x direction and y direction, the same wavelet transform is used for both directions. The final 2D filter is a tensor product of two 1D wavelet filters. From our experience of processing image data, we know that this kind of tensor product is an efficient way to deal with matrix-type data. There are other alternatives, such as non-separable wavelet filters, that can be used to decompose data, but there are no suitable non-separable wavelet filters that exist as far as we know. Therefore, a tensor product wavelet filter is used in processing DEM data. This approach is verified to satisfy the requirements.

As in processing image data, we developed a multiresolution representation for DEM data. When hosting GIS services over the Internet, this multiresolution format allows different resolution data to transfer separately and be integrated on the client side based on a user's request. The new data format for DEMs is called Wavelet Digital Elevation Model (WDEM). The next section defines the storage and transmission format for WDEM.

5.2 Wavelet Digital Elevation Models

Based on the features of DEMs, we apply discrete wavelet transforms on original DEMs and define the Wavelet Digital Elevation Model (WDEM) format. In this chapter, a simple threshold compression technique is used to show the compression capacity of the WDEM format.

The basic framework for storing WDEM data is a multiresolution framework. The control information in a DEM need to be preserved. The number of total rows and columns, the coordinates of the bottom left corner point, cell size, and null data value are in the same form as in the DEM. The 2D array in a DEM is decomposed into a multiresolution representation. The initial level is the coarsest representation, then a series of wavelet-coefficients are stored. These wavelet coefficients can be combined with the initial level to reconstruct different resolution representations. The most detailed resolution view is the same as the original DEM data if there is no lossy compression in the computational process. When lossy compression is allowed, then the most detailed data gives the best approximation of the original DEM data among all resolutions. The error of this approximation depends on the extent of lossy compression. The storage format, which is also the transmission format, for a WDEM is:

preserved data from a DEM:	*nrows*, *ncols*, (x_0, y_0), cell size (c), null data value (NDV)
number of resolution:	*nres*
the initial level:	*nrows1*, *ncols1*, $z_{[nrows1][ncols1]}$
wavelet coefficients:	3 compressed 2D arrays for each level (cV, cH, cD), totally *nres* levels

Table 5.2: Storage format for WDEM

In this representation, the wavelet coefficients for every level are organized into three compressed 2D arrays, which represent vertical details, horizontal details, and diagonal details. The size of these arrays in different levels are different. Generally the size of the finer level wavelet coefficients is quadruple the size of the neighboring coarser level coefficients if there is no compression. This relationship may be slightly

modified if the size of the original DEM data is not a power of 2. This will be discussed in section 5.3. The wavelet coefficients are compressed by a combination of lossless compression techniques (such as run-length coding, Huffman coding, or arithmetic coding) and lossy compression techniques (such as quantization which includes thresholding).

5.3 Implementation and Results

When implementing the WDEM format, we need to consider three issues. The first one is object wrapping. Because the format is defined for transmitting data over networks, it is more convenient to encapsulate an array as an object to transfer it. Objects are transmitted efficiently using the Java serialization protocol. The second issue is dealing with boundaries. Since the row size and column size of most DEMs are not powers of two, we generally need to extend the processed data at the boundaries in order to apply the discrete wavelet transform. There are different approaches to extend the data. Mirroring the data is used to extend the data in our implementation since this method does not need extra storage. This method is also easy implemented and it gives good results when symmetric filters are used [47]. The third issue is to use the lifting scheme to implement discrete wavelet transforms. In this way, generally a half the computation can be saved, although integer transforms cannot be used as in processing images (see Chapter 4).

As a starting point, we use the 2D Haar wavelet to process a DEM. The original DEM is shown in Figure 5-1. The Haar wavelet result is shown in Figure 5-2. A 2% threshold on the wavelet coefficients is used to compress the data. The 2% threshold compression means that we use 2% of the range of height values in a DEM as a threshold; all the wavelet coefficients with magnitudes less than this threshold will be replaced by zeros. This is a simple method to explore the compression potential of a data set.

From Figure 5-2, one can see that using the simple Haar wavelet and a threshold compression scheme produces a lot of block artifacts in the 3D view. This is because of

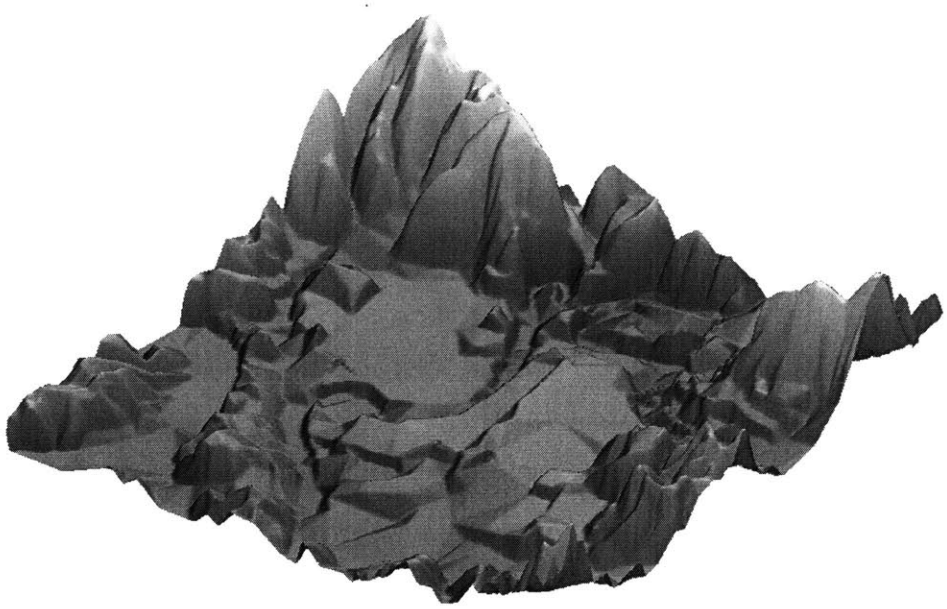


Figure 5-1: Original DEM

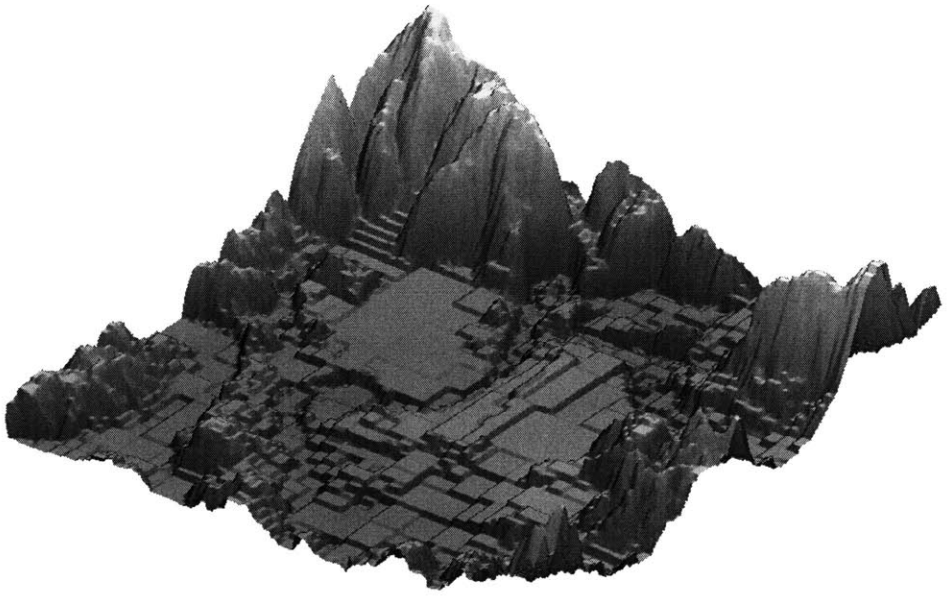


Figure 5-2: Haar wavelet processed, 2% threshold compression, 5-level decomposition

the discontinuity in the Haar wavelet and the simplicity of the threshold compression. There are different methods to improve these results. One way is to use better quantization methods than the threshold compression, such as uniform quantization. Another way is to use higher order wavelet filters to transform the data. In this section, we explore different wavelet filters to see the results in processing DEM data. Figure 5-3, Figure 5-4, and Figure 5-5 show the results from different wavelet filters. We can see that using the binlet 5/3 filter gives a reasonable good result. If a more carefully designed quantization method is used, the result should satisfy general purpose requirements. Table 5.3 show a more detailed comparison using different wavelet filters. Notice that visual quality is used as a complement to PSNR for evaluating the goodness of a 3D view. A visual quality of 5 represents the best quality, which has almost the same visual effect as the original data.

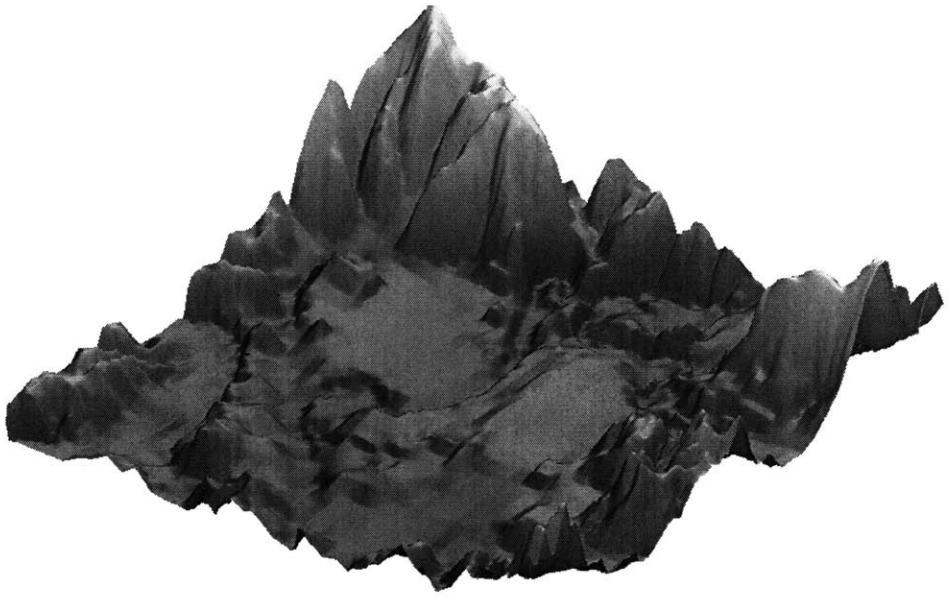


Figure 5-3: Binlet 5/3 wavelet processed, 2% threshold compression, 5-level decomposition

5.4 Spatial Analysis Based on WDEM

As we mentioned in the first section of this chapter, many different kinds of spatial analysis are based on DEMs. When we transform DEMs to WDEMs, an obvious

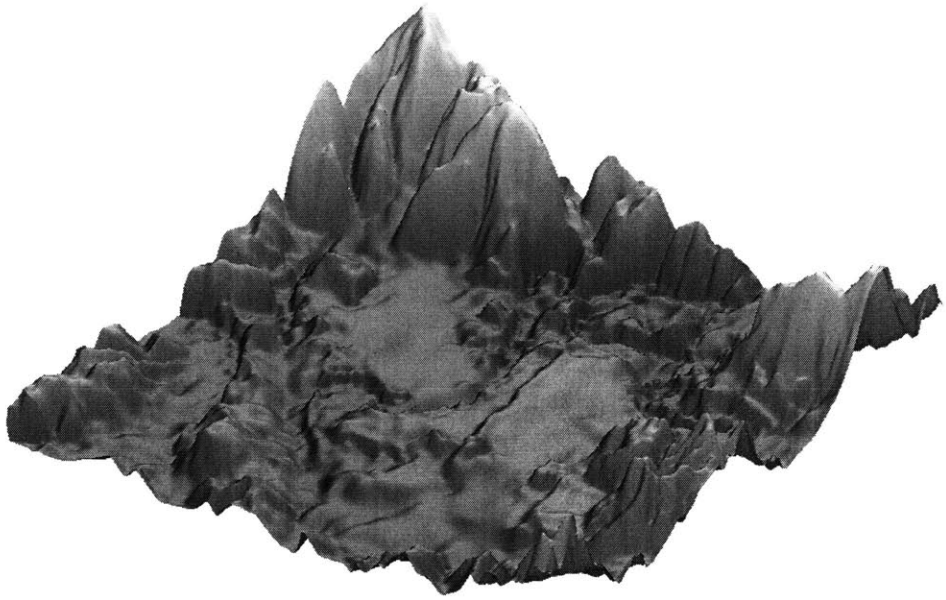


Figure 5-4: Bior7/5 wavelet processed, 2% threshold compression, 5-level decomposition

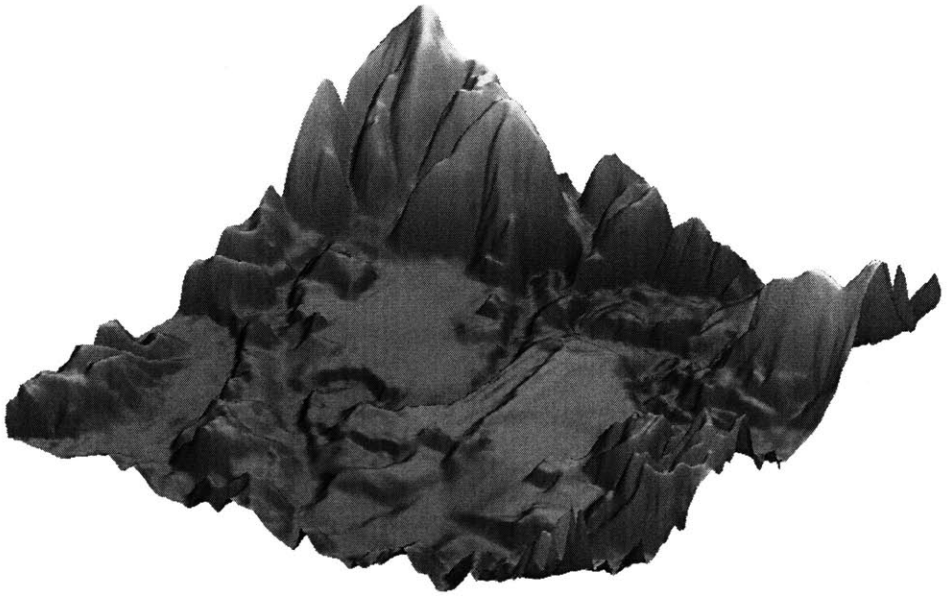


Figure 5-5: Bior9/7 wavelet processed, 2% threshold compression, 5-level decomposition

	Haar wavelet			Binlet 5/3 wavelet		
Threshold	0.1%	2%	5%	0.1%	2%	5%
Compression Rate (%)	37	89.54	98.28	60.15	96.65	99.22
Compression Ratio (*:1)	1.59	9.56	58.14	2.51	29.85	128.21
PSNR	70.47	39.19	31.11	70.28	41.85	34.31
Visual quality (0-5)	5	2	1	5	3.5	2.5
	Bior7/5 wavelet			Bior9/7 wavelet		
Threshold	0.1%	2%	5%	0.1%	2%	5%
Compression Rate (%)	57.66	93.29	97.14	57.01	94.12	97.37
Compression Ratio (*:1)	2.36	14.90	34.97	2.33	17.01	38.02
PSNR	67.04	41.85	33.85	70.71	47.98	42.7
Visual quality (0-5)	5	4	3.5	5	4.5	4.5

Table 5.3: Results using different wavelets for WDEMs

question that arises is whether the WDEM format is more efficient for spatial analysis. Therefore, in this section we will discuss spatial analysis on WDEMs in more detail.

Besides of the advantage of progressive transmission over networks, the WDEM format is also suitable for spatial analysis. The best reasoning is to give some examples. The first example is the *polygon-from-line* operation. Let us think about the following problem: which counties in Massachusetts does the freeway I-95 pass through? This is a typical problem for urban planning researchers. If using DEM data, we need to operate on all cells to identify whether I-95 passes through them. While using a WDEM, we can solve the intersection problem starting from the coarsest level data. In the coarsest level data, a very small problem can be solved quickly. After that, only those cells intersecting with I-95 in the coarsest level need to be further checked in finer resolutions. In this way, the hierarchical structure in a WDEM is used to reduce the problem size.

The second example is the *polygon-adjacency* problem. A typical question is which towns have common boundaries with Boston. A similar approach to that used in the last example can be applied here. If using a DEM, adjacency checking need to be done for each cell in the DEM. However, using a WDEM, several small cell-adjacency problems need to be solved because we can use the following reasoning to check cell adjacency at a finer resolution: the adjacency in a coarser level imply the *possible* adjacency in a finer level, the nonadjacency in a coarser level will *determinately*

implies the nonadjacency in a finer level. This implication will reduce the search times for this kind of spatial selection problem.

The third example is the *point-in-polygon* problem. A typical problem in the GIS domain is to find all the gas stations in the buffer of a freeway and in a specified region. Then firstly, we need to know which cells are in both the buffer of the freeway and the specified region. Secondly, we need to choose all the gas stations in the cells satisfying the previous requirements. This may need a large amount of comparisons. In a WDEM, computation time can be saved because one can use a coarser level data to do the comparison first and then consider more details afterwards.

The fourth example is the *spatial statistics* problem. An agriculture researcher may need to know all the areas whose average heights are below 300 meters in order to obtain information about the areas that are good for some kind of plant to grow in. In this situation, because some wavelet transforms give the average heights automatically in a multiresolution format, the corresponding coarse resolution data can be used directly to identify the areas satisfying the requirements.

5.5 Comparison of WDEM with Raster Pyramid Format

As the final word for WDEM, we compare WDEM with the raster pyramid format that is introduced in ArcInfo [52]. In the raster pyramid format, downsampling operations are used to obtain a hierarchical representation of DEMs. Although this is a simple way to implement multiresolution visualization, it needs extra memory to store each resolution. Furthermore, a coarser resolution in the raster pyramid format may not provide a good approximation of the original data because a lot of features are lost in the down-sampling operation. In the WDEM format, a coarser resolution is a good approximation of the original data because it exploits the correlation between adjacent cells. Therefore, the WDEM format will be a better alternative to the raster pyramid format.

Chapter 6

Second Generation Wavelets and Subdivision

In the previous three chapters, we have described the first generation wavelet theory and its applications in processing image maps and DEMs. First generation wavelets deal with data in a regular pattern or setting (uniformly spaced). However, most multi-dimensional data in GIS and 3D geometric modelling do not occur in this regular setting. They are often discretized as triangles and quadrilaterals. Although a raster-like 3D grid can be created to model a 3D object, uniformly spaced sampling is not preferred because it is not flexible and generates much redundant data. Therefore, more general multiresolution techniques are required to process data in an irregular setting. The construction of first generation wavelets is based on the shift invariance and scale invariance of the scaling functions and wavelet functions, which is related to the regular setting of the data. When constructing wavelets on data in an irregular setting, we cannot use the same approach that we used when constructing first generation wavelets. Fortunately, the lifting scheme inspires us to view wavelets from a different viewpoint, and it also provides a direct way to construct wavelets on irregular setting data. These wavelets are called second generation wavelets. This chapter provides a brief introduction to second generation wavelets. Applications on curves and surfaces will be discussed in following chapters.

6.1 Reexamination of the Lifting Scheme

From Chapter 4, we know that the lifting scheme implementation is more efficient than the traditional Mallat algorithm. It actually decomposes the polyphase matrix into elementary matrices. Here, we present the lifting scheme as an alternative way to construct wavelets.

The traditional filter bank approach is to partition data into two sub-sequences and apply a lowpass filter and a highpass filter on these two sub-sequences respectively. We repeat the above process on the output of the lowpass filter until a sufficiently deep multiresolution hierarchy is obtained. In the frequency domain, this process decomposes data into different subbands. However, when implementing the transform by the lifting scheme, this process is described as a sequence of different predicting and updating filter operations, which transfer information between the two partitions. This is shown in Figure 6-1.

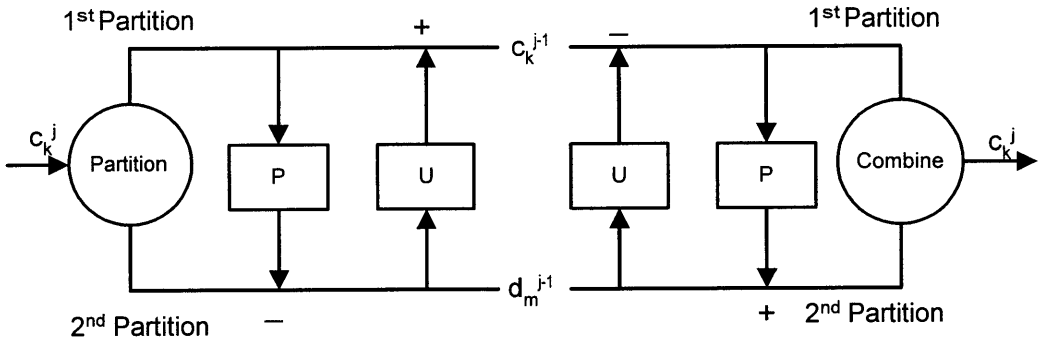


Figure 6-1: Predictors and updaters in the lifting scheme

The partitions are written as:

$$\mathcal{K}^{j+1} = \mathcal{K}^j \cup \mathcal{M}^j \ , j = 0, 1, \ldots, N \ , \tag{6.1}$$

where N refers to the finest scale, where the data is in its original and finest form. Notice that because of the partition operation, $\mathcal{K}^j$ and $\mathcal{M}^j$ have no overlap (*disjoint sets*). Therefore, equation (6.1) can also be written as:

$$\mathcal{M}^j = \mathcal{K}^{j+1} \setminus \mathcal{K}^j \ , \ j = 0, 1, \ldots, N \ . \tag{6.2}$$

In one dimension, the partitioning operation can be interpreted as separating the signal into even- and odd- indexed coefficients. We can iteratively apply the above process on $\mathcal{K}^j$ to obtain a nested group, which has the following structure:

$$\begin{aligned} &\mathcal{K}^0 \subset \mathcal{K}^1 \subset \mathcal{K}^2 \subset \cdots \subset \mathcal{K}^N \\ &\mathcal{K}^i \cup \mathcal{M}^i = \mathcal{K}^{i+1},\ i = 0, \ldots, N-1 \end{aligned} \tag{6.3}$$

Note that the superscripts are used to represent different resolutions (larger numbers represent finer resolutions). $\mathcal{K}^N$ denotes the finest representation of the data. $\mathcal{K}^N$ can be partitioned into $\mathcal{K}^{N-1}$ and $\mathcal{M}^{N-1}$; then $\mathcal{K}^{N-1}$ can be partitioned into $\mathcal{K}^{N-2}$ and $\mathcal{M}^{N-2}$, and so on, until $\mathcal{K}^1$ is partitioned into $\mathcal{K}^0$ and $\mathcal{M}^0$. Based on this nested (or *hierarchical*) structure of the partition, one can construct wavelets and approximations of the data.

In Figure 6-1, if considering only one predictor and one updater for simplicity, the output of the DWT can be written as:

$$\bar{c}_{\mathcal{K}^j} = c_{\mathcal{K}^{j+1}}(\mathcal{K}^j)\ , \tag{6.4}$$

$$\bar{d}_{\mathcal{M}^j} = c_{\mathcal{K}^{j+1}}(\mathcal{M}^j)\ , \tag{6.5}$$

$$\bar{c}_{\mathcal{M}^j} = P_j(\bar{c}_{\mathcal{K}^j})\ , \tag{6.6}$$

$$d_{\mathcal{M}^j} = \bar{d}_{\mathcal{M}^j} - \bar{c}_{\mathcal{M}^j}\ , \tag{6.7}$$

$$\Delta c_{\mathcal{K}^j} = U_j(d_{\mathcal{M}^j})\ , \tag{6.8}$$

$$c_{\mathcal{K}^j} = \bar{c}_{\mathcal{K}^j} + \Delta c_{\mathcal{K}^j}\ . \tag{6.9}$$

Notice that P_j and U_j are two symbolic operators. For a linear predictor and updater, we can write the above formulas in a fully indexed notation as:

$$\bar{c}_{j+1,m} = \sum_{k \in \mathcal{K}^j} p_{j,m,k} c_{j+1,k} \quad ,\ d_{j,m} = c_{j+1,m} - \bar{c}_{j+1,m}\ , \quad \forall m \in \mathcal{M}^j\ ; \tag{6.10}$$

$$\Delta c_{j+1,k} = \sum_{m \in \mathcal{M}^j} u_{j,k,m} d_{j,m} \quad ,\ c_{j,k} = c_{j+1,k} + \Delta c_{j+1,k}\ , \quad \forall k \in \mathcal{K}^j\ . \tag{6.11}$$

The IDWT reverses the process by merging the $\mathcal{K}^j$ and $\mathcal{M}^j$ parts. In first gener-

ation wavelets, $p_{j,m,k}$ and $u_{j,k,m}$ are independent of the scale parameter j and the position parameters k and m. They depend only on the relative position of k and m. Therefore, the equivalent wavelet filters $\{\tilde{l},\ l,\ \tilde{h},\ h\}$ in first generation wavelets are also independent of scale and position. If wavelet filters do not depend on scale, they are called "stationary filters"; if they do not depend on position, they are called "uniform filters". However, for the general case, the scale index and the position index are needed for expressing filters, which results in generalized wavelets. These wavelets are called second generation wavelets.

From the above discussion, we can see that the lifting scheme can be used to construct generalized wavelets, which do not hold the scaling invariant and the shifting invariant properties. Wavelet coefficients become the difference between finer level data and the estimation of them based on partitioned data. Coarser level data becomes finer level data plus an update based on the wavelet coefficients. All these operations can be designed and implemented in time/space domain, without explicit frequency domain information. This overcomes the biggest obstacle for processing irregular setting data by wavelet transforms.

6.2 Extended MRA

Since the scaling invariant and the shifting invariant laws do not hold in a general wavelet transform, the concept of multiresolution analysis—MRA—defined in Chapter 3 needs to be extended as well. A formal definition of the extended MRA is [49]:

1. $\cdots \subset V_{-1} \subset V_0 \subset V_1 \subset \cdots$;

2. $\bigcap_{j\in\mathcal{Z}} V_j = 0 \quad \overline{\bigcup_{j\in\mathcal{Z}} V_j} = L^2(\mathcal{R})$;

3. V_j has a stable basis (Riesz basis) $\{\phi_{j,k}(t)|k \in \mathcal{K}^j\}$.

Here, $\mathcal{K}^j$ is a general position index set, which satisfies $\mathcal{K}^j \in \mathcal{K}^{j+1}$. The complement subspace of V_j in V_{j+1} is defined as W_j. Its basis functions are $\{\psi_{j,m}(t)|m \in \mathcal{M}^j\}$.

This definition is in a very general sense. It only requires a sequence of nested subspaces that have stable bases. This includes non-stationary and nonuniform wavelet transforms. However, for most applications, some internal static structures in the data need to be explored, which requires some similarities in the bases of these nested subspaces $\{V_j\}$. Otherwise the multiresolution structure does not have many advantages for analysis and the computation will be expensive.

6.3 Second Generation Wavelets

The extended MRA concept provides the theoretical foundation to develop wavelets on an irregular setting. Second generation wavelets are born in such a need. They are the wavelets on irregular setting data. Because of this, the Fourier transform cannot be used to construct second generation wavelets although it is the major tool to construct first generation wavelets. A new theory must be put forward to fit in this position. The lifting scheme, which uses only time/space domain information, eventually becomes a powerful tool to construct second generation wavelets. The basic idea comes from two sides. One is from the work of Michael Lounsbery, Tony De Rose, and Joe Warren [35, 36]. The other is from the work of David Donoho [17]. Later on, Sweldens [49] formalized the theory for second generation wavelets.

6.3.1 Basic Formulas

The basic idea can be explained from equations (6.12)$\sim$(6.13). In first generation wavelets, we derive the basis functions $\phi_{j,k}(t)$ and $\psi_{j,k}(t)$ from the same scaling function $\phi(t)$, which are constrained by the refinement equation. The mother wavelet function $\psi(t)$ also comes from $\phi(t)$ by the wavelet equation. The dyadic scaling and shifting invariance are beautiful relationships, but they are destroyed in an irregular setting. More general refining equations replace them.

$$\phi_{j,k}(t) = \sum_{m \in \mathcal{K}^{j+1}} l_{j,k,m} \phi_{j+1,m}(t) \ , \tag{6.12}$$

$$\psi_{j,k}(t) = \sum_{m \in \mathcal{K}^{j+1}} h_{j,k,m} \phi_{j+1,m}(t) \ . \tag{6.13}$$

The above refining equations can be thought of as nonuniform scaling and shifting relationships. $\phi_{j,k+1}(t)$ can be thought of as a nonuniform shift of $\phi_{j,k}(t)$ by a step. Notice that the step size for each k is different. $\phi_{j+1,k}(t)$ can be thought of as a nonuniform scaling of $\phi_{j,k}(t)$ by a factor, which is also different for each j and k. The rule behind this is that they are the same kind of functions and defined on grids which are not necessarily equally spaced but have some internal structure.

In first generation wavelets, we have

$$l_{j,k,m} = l[m - 2k] \ , \tag{6.14}$$

$$h_{j,k,m} = h[m - 2k] \ . \tag{6.15}$$

Then the refining equations become the scaling and shifting invariant relationships discussed in Chapter 3.

One simple example of a refining equation on nonuniform grids is a hat function defined on nonuniform grids. The refinement equation for a hat function can be

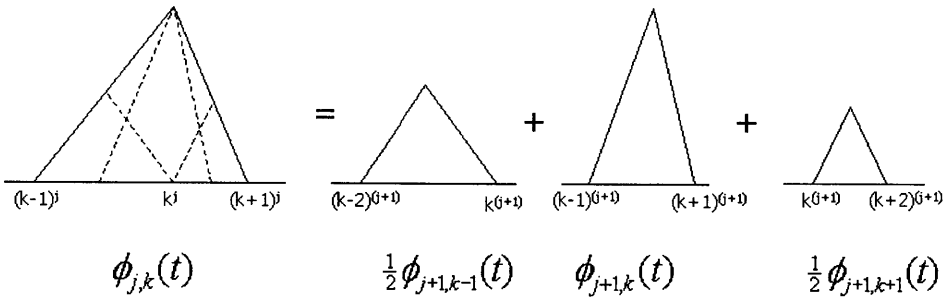


Figure 6-2: Nonuniform hat function

written as:

$$\phi_{j,k}(t) = \frac{1}{2}\phi_{j+1,k-1}(t) + \phi_{j+1,k}(t) + \frac{1}{2}\phi_{j+1,k+1}(t) \ . \tag{6.16}$$

In a general biorthogonal case, scaling functions and wavelet functions in primary and dual subspaces have the same relationships as before. The biorthogonality

condition, then, becomes:

$$< \psi_{j,k}, \tilde{\psi}_{j',m'} >= \delta[j - j'] \, \delta[m - m'] \, . \tag{6.17}$$

Summarizing all the constraints in a biorthogonal wavelet transform, the scaling functions, wavelets, dual scaling functions, and dual wavelets have the following relationships:

$$\begin{aligned}
< \tilde{\phi}_{j,k}, \phi_{j,k'} > &= \delta[k - k'] \, , & (6.18)\\
< \tilde{\psi}_{j,m}, \psi_{j,m'} > &= \delta[m - m'] \, , & (6.19)\\
< \tilde{\phi}_{j,k}, \psi_{j,m} > &= 0 \, , & (6.20)\\
< \tilde{\psi}_{j,m}, \phi_{j,k} > &= 0 \, . & (6.21)
\end{aligned}$$

Notice that equation (6.17) is included in the above equations. These constraints can also be written as relationships among the corresponding biorthogonal filters.

$$\begin{aligned}
\sum_{n \in K^{j+1}} l_{j,k,n} \tilde{l}_{j,k',n} &= \delta[k - k'] \, , & (6.22)\\
\sum_{n \in K^{j+1}} h_{j,m,n} \tilde{h}_{j,m',n} &= \delta[m - m'] \, , & (6.23)\\
\sum_{n \in K^{j+1}} h_{j,m,n} \tilde{l}_{j,k,n} &= 0 \, , & (6.24)\\
\sum_{n \in K^{j+1}} l_{j,k,n} \tilde{h}_{j,m,n} &= 0 \, , & (6.25)
\end{aligned}$$

where $j \in \{0, 1, \ldots, J-1\}$, $m \in M^j$, $k \in K^{j+1}$. For a discrete time signal $g = \{c_{J,k} | k \in K^J\}$, the DWT is:

$$c_{j,k} = \sum_{n \in K^{j+1}} \tilde{l}_{j,k,n} c_{j+1,n} \; k \in K^j, \; j = J-1, J-2, \ldots, 0 \, , \tag{6.26}$$

$$d_{j,m} = \sum_{n \in K^{j+1}} \tilde{h}_{j,m,n} c_{j+1,n} \; m \in M^j, \; j = J-1, J-2, \ldots, 0 \, ; \tag{6.27}$$

the IDWT is:

$$c_{j+1,n} = \sum_{k \in K^j} l_{j,k,n} c_{j,k} + \sum_{m \in M^j} h_{j,m,n} d_{j,m} \ , \qquad (6.28)$$
$$n \in K^{j+1} \ , \ j = 0, 1, 2, \ldots, J-1 \ .$$

6.3.2 Construction by the Lifting Scheme

Comparing to the lifting scheme in equations (6.6)~(6.9), we can construct second generation wavelets using the lifting technique. The predictor and updater in the lifting scheme have the following relationships with the lowpass filters and the highpass filters.

$$\tilde{h}_{j,m,n} = \tilde{h}^{\text{old}}_{j,m,n} - \sum_{k \in K^j} p_{j,k,m} \tilde{l}^{\text{old}}_{j,k,n} \ , \forall n \in K^{j+1} \ , m \in M^j \ , \qquad (6.29)$$

$$\tilde{l}_{j,k,n} = \tilde{l}^{\text{old}}_{j,k,n} + \sum_{m \in M^j} u_{j,m,k} \tilde{h}_{j,m,n} \ , \forall n \in K^{j+1} \ , k \in K^j \ . \qquad (6.30)$$

Here, $\tilde{l}^{\text{old}}_{j,k,n}$ and $\tilde{h}^{\text{old}}_{j,m,n}$ correspond to the partition operation. Without the lifting step, the IDWT is simply the inverse of partition operation:

$$c_{j+1,n} = \sum_{k \in K^j} l^{\text{old}}_{j,k,n} c_{j,k} + \sum_{m \in M^j} h^{\text{old}}_{j,m,n} d_{j,m} \ , \forall n \in K^{j+1} \ , \qquad (6.31)$$

where $l_{j,k,n}$ and $h_{j,m,n}$ correspond to the merge operation.

$$l^{\text{old}}_{j,k,n} = \delta[n-k] \ , \forall k \in K^j \ , \qquad (6.32)$$

$$h^{\text{old}}_{j,m,n} = \delta[n-m] \ , \forall m \in M^j \ . \qquad (6.33)$$

Through the lifting scheme, the new lowpass and highpass filters for the IDWT are:

$$l_{j,k,n} = l^{\text{old}}_{j,k,n} + \sum_{m \in M^j} p_{j,k,m} h^{\text{old}}_{j,m,n} \ , \qquad (6.34)$$

$$h_{j,m,n} = h^{\text{old}}_{j,m,n} - \sum_{k \in K^j} u_{j,m,k} l_{j,k,n} \ . \qquad (6.35)$$

Although here l^{old}, h^{old}, $\tilde{l}^{\text{old}}$, and $\tilde{h}^{\text{old}}$ express merge and partition operations, which are naturally biorthogonal, all the above formulas are correct for constructing new biorthogonal filters through the lifting scheme based on any set of old biorthogonal filters.

All the above relationships can also be written for scaling functions and wavelets.

$$\tilde{\psi}_{j,m} = \sum_m \tilde{h}_{j,m,n}\tilde{\phi}^{\text{old}}_{j+1,n} = \tilde{\psi}^{\text{old}}_{j,m} - \sum_{k\in K^j} p_{j,k,m}\phi^{\text{old}}_{j,k} \ , \tag{6.36}$$

$$\tilde{\phi}_{j,k} = \sum_n \tilde{l}_{j,k,n}\tilde{\phi}^{\text{old}}_{j+1,n} = \tilde{\phi}^{\text{old}}_{j,k} + \sum_{m\in M^j} u_{j,m,k}\tilde{\psi}_{j,m} \ , \tag{6.37}$$

$$\phi_{j,k} = \sum_n l_{j,k,n}\phi^{\text{old}}_{j+1,n} = \phi^{\text{old}}_{j,k} + \sum_{m\in M^j} p_{j,k,m}\psi^{\text{old}}_{j,m} \ , \tag{6.38}$$

$$\psi_{j,k} = \sum_n h_{j,m,n}\phi^{\text{old}}_{j+1,n} = \psi^{\text{old}}_{j,m} - \sum_{k\in K^j} u_{j,m,k}\phi_{j,k} \ . \tag{6.39}$$

6.4 Subdivision and Lifting

Second generation wavelets are constructed on irregular setting data. We can build wavelets for fully unstructured data, but such a transform needs space to store all the spatial information and the filter coefficients for every point at every scale, which is not efficient for most applications. Currently, processing a three-dimensional object is the most challenging application. Surface data defined on a triangular mesh cannot be processed using the tensor product wavelet filters. The lifting scheme tells us that the key steps to construct second generation wavelets (starting from a good existing biorthogonal wavelet filter) are: finding a good predictor to do prediction, and finding a good updater to achieve a better coarse representation.

6.4.1 Lazy Filter and Partition

For irregular setting data, the simplest and most convenient existing biorthogonal filter is the lazy filter [43], which essentially partitions the original data. In the 1D case, when combined with the down sampling operation, it partitions the data into odd-indexed data and even-indexed data. Figure 6-3 shows this operation. In this

figure, z^{-1} means right-shift (delay) whole signal one index.

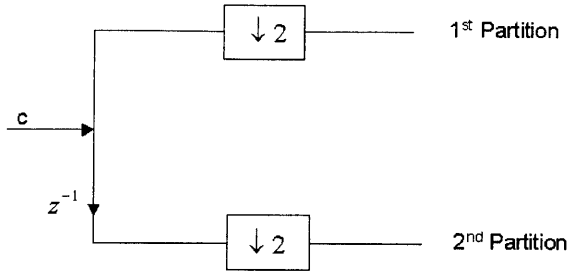


Figure 6-3: the Lazy filter

This idea is easily extended to higher dimensional irregular setting data, where the original data is partitioned to different subsets. The data at scale j can be partitioned into two sets with indices K^{j-1} and M^{j-1}. Then a predictor is used to estimate the data on M^{j-1} by using the data on K^{j-1}. The differences of the estimations and the original data are called the wavelet coefficients. After that, an updater can be used to improve the data quality on K^{j-1} according to a suitable quality measurement criterion.

6.4.2 Prediction

In an irregular setting, a good predictor is often obtained from a subdivision scheme. Generally, a stationary subdivision scheme can achieve good results. The algorithm is efficient and the result has reasonable accuracy. Subdivision is an iterative process that uses masks to create new data based on coarser resolution data. It does not require data in a regular setting. Subdivision can be thought of as iteratively applying a synthesis lowpass filter on coarser resolution data in the second generation wavelet framework. It follows that the subdivision scheme matches the prediction filter in the lifting scheme. Subdivision is not a new concept. It has been widely used in computer geometry design. It also has close relationship with fractals. The most fundamental work started in 1959 by P. de Casteljan, and later by P. Bézier in 1966 [21]. A good 1D example is a Bézier curve or a B-spline. Using subdivision, a complex curve or surface can be constructed from several basic points and a simple iterative process.

This is similar to building an approximation using only the lowpass synthesis filter in a wavelet transform.

For surface data, triangles and quadrilaterals are the most popular patterns used in subdivision. This is consistent with finite element analysis on real structures and other geometric design tools. The most useful subdivision scheme for a triangular configuration is based on quaternary triangulation, which divides a triangle into four similar triangles. Figure 6-4(a) shows one step of the subdivision. Generally, a subdivision scheme can be categorized into two types: interpolation and approximation. In an interpolation scheme, once a new vertex is inserted into the configuration, it will not be changed in the subsequent subdivision. However, an approximation scheme will update the existing vertices in the current configuration after inserting new points. For the triangular setting, Loop [34] developed an approximation subdivision scheme, which can create C^2 continuous surfaces. It uses quadtree triangulation. The mask for evaluating the newly inserted points is shown in Figure 6-4(b). The mask for re-evaluating the existing points is shown in Figure 6-4(c). Dyn et al. [19] developed

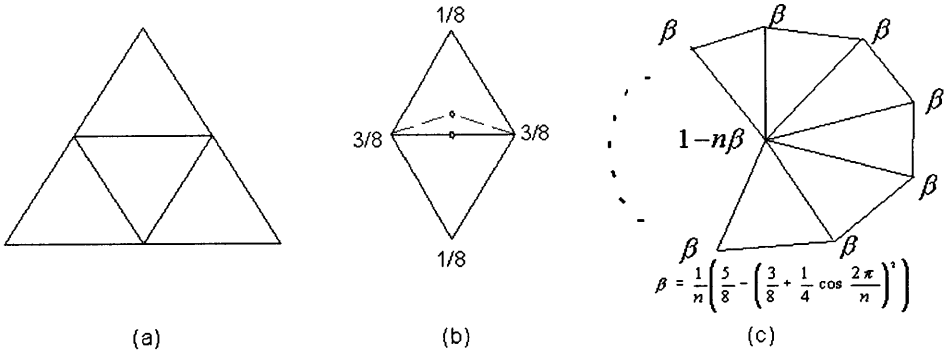


Figure 6-4: Loop scheme

an interpolation subdivision scheme, which is called the Butterfly scheme. It uses quaternary triangulation as well. The evaluation mask is shown in Figure 6-5. For a quadrilateral configuration, several schemes exist, such as Doo-Sabin, Catmull-Clark, Kobbelt etc. Because we will not use quadrilateral configurations, those schemes are not discussed here.

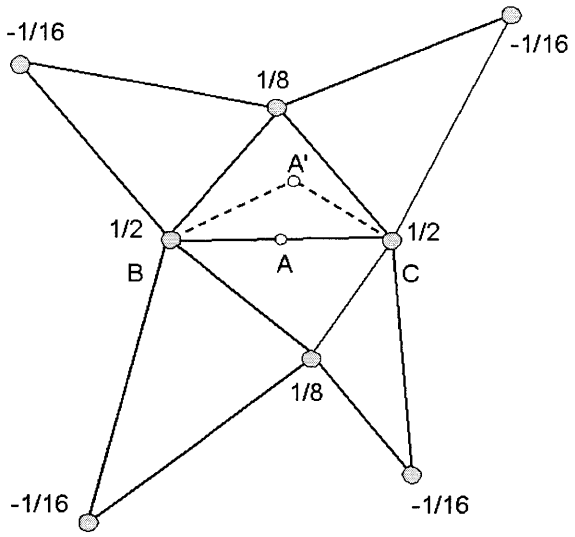


Figure 6-5: Butterfly scheme

6.4.3 Updating and Interpolation

After the prediction step, an updater can be used to get a better representation on coarser level data. Since the wavelet coefficients have already been produced after the prediction step, an updater is applied on these wavelet coefficients to update the other partition, which leads to the coarse representation. In some GIS applications, the interpolation feature is desired, which means that a finer scale representation is obtained only by adding new points to the coarse scale representation without changing the coarse data. This feature provides better perception of surface point positions. In this case, the updater needs to be dropped in the lifting scheme framework because an updater will change the data from the coarser representation. Therefore, the lifting scheme for an interpolation wavelet filter becomes simplified as illustrated in Figure 6-6.

Of course, there are some applications for which good approximation properties are more important than the interpolation feature, such as image processing and some visualization modeling. In those cases, updaters are generally designed to improve the vanishing moments of the wavelets. Schröder and Sweldens [43] have done some work for data on a spherical geometry. Other updating rules also exist. For example, Stollnitz et al. [46] wanted the wavelets to be orthogonal to the scaling functions as

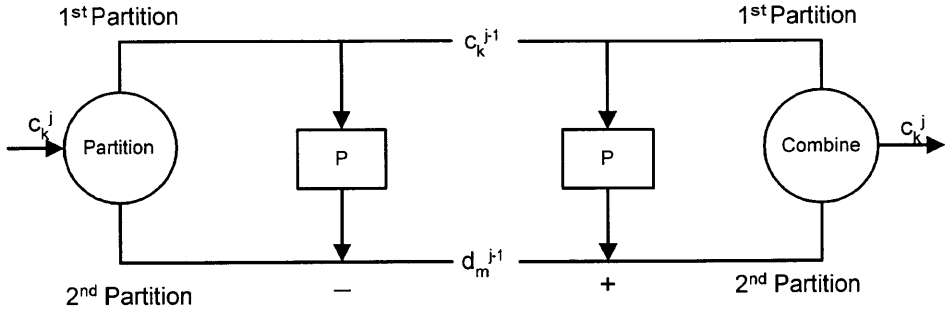


Figure 6-6: Interpolation wavelets constructed by lifting scheme

closely as possible. This is similar to the energy compaction rule.

Notice that although subdivision has a close relationship with second generation wavelets, they are not exactly the same. The wavelet framework provides two or more channels of information, where subdivision only provides information on the self-similar channel.

Chapter 7

Wavelet Arc (WArc) Data Representation

Through the discussion of second generation wavelets in the last chapter, we have shown how second generation wavelets can be constructed on irregular setting data. This construction is useful for processing curves and surfaces. In this chapter we will focus on curve representations and the next chapter will focus on triangulated surface representations.

7.1 Arc Format

Currently, most popular GIS software processes curves in the Arc format. This includes lines, circular arcs, elliptical arcs, polygons, and polylines. All of these are called curvilinear features and can be represented by a sequence of points in space. In order to simplify our discussion, we think of an Arc as being composed of a sequence of points connected with lines. The related topological information, such as neighboring polygons, is stored with Arcs as well. Geodatabases are used to store and retrieve this information. Here we care more about curves themselves. So we do not consider topological information related to curves.

Table 7.1 shows the typical data in the Arc format. This data format is general for curves and polygons. Notice that segments are used to group points since different

Total segments	*nseg*
Numbers of points for each segment	*np1, np2, np3, ...*
Point coordinates	x_1, y_1, z_1 x_2, y_2, z_2 ...

Table 7.1: Arc format

segments may have different properties, such as measurement accuracy. This is an engineering method to manage large measurement data sets. Therefore, we will keep segments in our multiresolution data representation as well. The Arc format does not have a natural multiresolution representation. One simple way to represent the Arc in a multiresolution format is to downsample the data sequence. Iteratively downsampling a data sequence will lead to a multiresolution representation. However, this multiresolution representation is not a good one because the downsampled version is not a good approximation of the original data. Therefore, improved multiresolution techniques are needed to obtain a better representation. Second generation wavelets are used in this chapter to construct a multiresolution representation for curves.

7.2 Wavelet Arc Format

7.2.1 A Simple Example

In order to better understand the construction of wavelets on nonuniformly spaced data, a simple example is given below to explain how to use the lifting scheme to construct a multiresolution representation of curves. Figure 7-1 shows this example.

In order to simplify the notation for constructing second generation wavelets, we will not use the index version of discrete wavelet transforms, such as in equations (6.26) and (6.27). Instead, we use the symbolic version of the lifting scheme as shown in equations (6.6)~(6.9). A simplified version is presented here for convenience.

$$\begin{aligned} d_{j,m} &= c_{j+1,m} - P(c_{j+1,k}) \\ c_{j,k} &= c_{j+1,k} + U(d_{j,m}) \end{aligned} \tag{7.1}$$

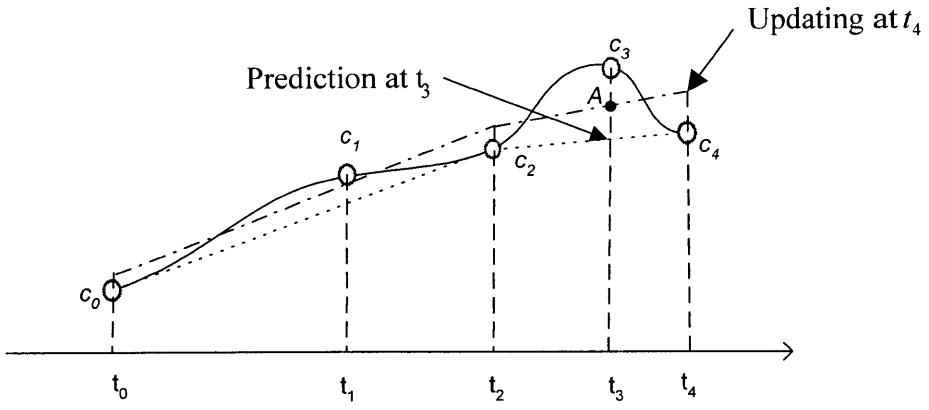


Figure 7-1: A simple example of second generation wavelets

Notice that the nested point sets are:

$$\begin{aligned} &\mathcal{K}^0 \subset \mathcal{K}^1 \subset \mathcal{K}^2 \subset \cdots \subset \mathcal{K}^N \,, \\ &\mathcal{K}^i \cup \mathcal{M}^i = \mathcal{K}^{i+1},\ i = 0, \ldots, N-1 \,. \end{aligned} \tag{7.2}$$

Points k and m in equation (7.1) belong to point sets $\mathcal{K}^j$ and $\mathcal{M}^j$ respectively.

In order to process the original curve represented by the point sequence $\{c_0, c_1, c_2, c_3, c_4\}$, three steps are taken to construct the wavelet transform. Given irregular data on points $\mathcal{K}^1 = \{t_0, t_1, t_2, t_3, t_4\}$, the one step partition and wavelet transform are:

$$\left\{ \begin{array}{l} c_{\mathcal{K}^1} = \{c_0, c_1, c_2, c_3, c_4\} \\ \mathcal{K}^0 = \{t_0, t_2, t_4\}, \mathcal{M}^0 = \{t_1, t_3\}, \mathcal{K}^1 = \mathcal{K}^0 \cup \mathcal{M}^0 \\ \bar{c}_{\mathcal{K}^0} = c_{\mathcal{K}^1}(\mathcal{K}^0), \bar{d}_{\mathcal{M}^0} = c_{\mathcal{K}^1}(\mathcal{M}^0) \\ \lambda_1 = \frac{t_1 - t_0}{t_2 - t_0}, \lambda_2 = \frac{t_3 - t_2}{t_4 - t_2} \\ d_{\mathcal{M}^0} = \bar{d}_{\mathcal{M}^0} - P(\bar{c}_{\mathcal{K}^0}) = \begin{bmatrix} c_1 \\ c_3 \end{bmatrix} - \begin{bmatrix} 1-\lambda_1 & \lambda_1 & 0 \\ 0 & 1-\lambda_2 & \lambda_2 \end{bmatrix} \begin{bmatrix} c_0 \\ c_2 \\ c_4 \end{bmatrix} \\ c_{\mathcal{K}^0} = \bar{c}_{\mathcal{K}^0} + U(d_{\mathcal{M}^0}) = \begin{bmatrix} c_0 \\ c_2 \\ c_4 \end{bmatrix} + \begin{bmatrix} 1-\lambda_1 & 0 \\ \lambda_1 & 1-\lambda_2 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} d_{t_1} \\ d_{t_3} \end{bmatrix} \end{array} \right. \tag{7.3}$$

Here P and U depend on $\mathcal{K}^0$ and $\mathcal{M}^0$ through parameters λ_1 and λ_2; therefore, it is a nonstationary and nonuniform transform. If this scheme is used to process a curve, all the λ's need to be stored. This may not result in an efficient representation. In this example, a linear predictor and a linear updater are used for P and U. The matrix for U is the transpose of the matrix for P. This corresponds to redistributing the wavelet coefficients (i.e. errors) in the same proportions as the data contributing to the prediction. It is easy to verify that the updater makes the norm defined in equation (7.4) achieve a smaller value than that without updating if λ_1 and λ_2 are 0.5. (Note that in equation (7.4), $c_{\mathcal{K}^0}(t_j)$ for $t_j \notin \mathcal{K}^0$ refers to the prediction value $P(c_{\mathcal{K}^0})$ at t_j. For example, point A in Figure 7-1 corresponds to $c_{\mathcal{K}^0}(t_3)$). For other values of λ_1 and λ_2, better updating filters need to be used. This is an example of optimization in the updating step. Equation (7.3) is called the analysis transform (only one step), which decomposes a finer resolution data into a coarser representation plus wavelet information. The inverse process, the synthesis transform, can be derived by changing the sign before the predictor and the updater.

$$\text{Norm} = \sum_{t_j \in \mathcal{K}^1} (c_{\mathcal{K}^1}(t_j) - c_{\mathcal{K}^0}(t_j))^2 \ . \tag{7.4}$$

7.2.2 Subdivision Matrix and Smoothness

From the previous example, we have seen that an important step for the lifting scheme is the prediction step. A good prediction will result in small wavelet coefficients. Therefore, we need to pay extra attention to this. A predictor is applied on one partition of the data to predict the other partition of the data. This filter operation really explores the correlation of two partitions (or two channels in the signal processing domain). If we use several predictors in series, they construct an iterative process, which leads to a subdivision operation. From here, we can see that a subdivision process defines a predictor. There are many subdivision schemes that exist for generating curves. A Bézier curve is a typical example. In this section, we use the four-point scheme to investigate the subdivision process.

The four-point scheme is an interpolation scheme which fits a C^1 continuous para-

metric cubic polynomial curve on four given points. Through a weighted summation of the coordinates of the four given points, we want to directly obtain the coordinates of the parametric middle point without explicitly evaluating the cubic polynomial. Figure 7-2 illustrates this.

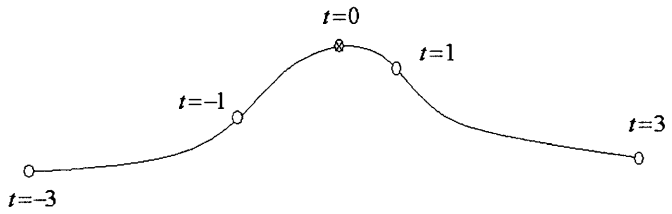


Figure 7-2: Four-point interpolation scheme

In Figure 7-2, for the given arbitrary four points with parameters $\{t = -3, -1, 1, 3\}$, we want to evaluate the point which corresponds to the parameter $t = 0$ in the parametric cubic polynomial defined on these four points. A simple process is to firstly fit a parametric cubic polynomial on these four points and then derive the formula with a weighted summation. The final format of the evaluation mask should have the following format:

$$c_0 = w_{-3}c_{-3} + w_{-1}c_{-1} + w_1c_1 + w_3c_3 \ .$$

A brief derivation is presented as below. Considering only the x coordinate, a cubic polynomial can be written as:

$$x(t) = at^3 + bt^2 + ft + g \ .$$

Then,

$$\vec{x}^0 = \begin{bmatrix} x(-3) \\ x(-1) \\ x(1) \\ x(3) \end{bmatrix} = \begin{bmatrix} -27 & 9 & -3 & 1 \\ -1 & 1 & -1 & 1 \\ 1 & 1 & 1 & 1 \\ 27 & 9 & 3 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ f \\ g \end{bmatrix} = G \begin{bmatrix} a \\ b \\ f \\ g \end{bmatrix} ,$$

where G is a symbolic representation of the coefficient matrix before the vector

$[a, b, f, g]^T$. We want to calculate $x(0)$, which is:

$$x(0) = a(0^3) + b(0^2) + f(0^1) + g = g \ .$$

Invert the matrix G, then we obtain:

$$G^{-1} = \frac{1}{48} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -3 & -3 & 3 \\ 1 & -27 & 27 & -1 \\ -3 & 27 & 27 & -3 \end{bmatrix} .$$

Then,

$$x(0) = g = \left(G^{-1}\right)_{row4} \vec{x}^0 = \frac{1}{16} \begin{bmatrix} -1 & 9 & 9 & -1 \end{bmatrix} \vec{x}^0 \ .$$

In the same way, we can obtain $y(0)$ and $z(0)$. Finally, we obtain:

$$c_0 = \frac{-1}{16} c_{-3} + \frac{9}{16} c_{-1} + \frac{9}{16} c_1 + \frac{-1}{16} c_3 \ . \tag{7.5}$$

This is the four-point subdivision scheme. The biorthogonal binary 9/7 filter (Binlet 9/7) uses this as the lowpass filter. It is an interpolation scheme because it directly gives a point on the parametric cubic polynomial curve; while other schemes may give the point on the curve using an iterative process, which leads to non-interpolation schemes.

Using the four-point scheme, we can get the characteristic matrix for this subdivision process. Equation (7.6) gives the expression. The symbolic expression can be written as in equation (7.7). The eigenvalues (equation (7.8)) and the eigenvectors (equation (7.9)) of the subdivision matrix S determine the convergence of the subdivision scheme and the smoothness of the final curves. The convergence condition of a subdivision scheme is that the maximum eigenvalue should be 1. The smoothness condition is determined by the tangent vector, which can be derived from the

subdivision matrix. A detailed discussion can be found in [55].

$$\begin{bmatrix} c_{i-3}^{j+1} \\ c_{i-2}^{j+1} \\ c_{i-1}^{j+1} \\ c_{i}^{j+1} \\ c_{i+1}^{j+1} \\ c_{i+2}^{j+1} \\ c_{i+3}^{j+1} \end{bmatrix} = \frac{1}{16} \begin{bmatrix} -1 & 9 & 9 & -1 & 0 & 0 & 0 \\ 0 & 0 & 16 & 0 & 0 & 0 & 0 \\ 0 & -1 & 9 & 9 & -1 & 0 & 0 \\ 0 & 0 & 0 & 16 & 0 & 0 & 0 \\ 0 & 0 & -1 & 9 & 9 & -1 & 0 \\ 0 & 0 & 0 & 0 & 16 & 0 & 0 \\ 0 & 0 & 0 & -1 & 9 & 9 & -1 \end{bmatrix} \begin{bmatrix} c_{i-3}^{j} \\ c_{i-2}^{j} \\ c_{i-1}^{j} \\ c_{i}^{j} \\ c_{i+1}^{j} \\ c_{i+2}^{j} \\ c_{i+3}^{j} \end{bmatrix}, \tag{7.6}$$

$$c^{j+1} = S\, c^{j}\,, \tag{7.7}$$

$$\lambda(S) = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{4} & \frac{1}{4} & \frac{1}{8} & -\frac{1}{16} & -\frac{1}{16} \end{bmatrix}, \tag{7.8}$$

$$v(S) = \begin{bmatrix} 0.3780 & 0.5669 & -0.6429 & -0.6429 & -0.6775 & 0 & 1 \\ 0.3780 & 0.3780 & -0.2857 & -0.2857 & -0.2008 & 0 & 0 \\ 0.3780 & 0.1890 & -0.0714 & -0.0714 & -0.0251 & 0 & 0 \\ 0.3780 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.3780 & -0.1890 & -0.0714 & -0.0714 & 0.0251 & 0 & 0 \\ 0.3780 & -0.3780 & -0.2857 & -0.2857 & 0.2008 & 0 & 0 \\ 0.3780 & -0.5669 & -0.6429 & -0.6429 & 0.6775 & 1 & 0 \end{bmatrix}. \tag{7.9}$$

7.2.3 Storage and Transmission Format for WArc

We have seen that subdivision filters are good candidates for predictors in the lifting scheme. The eigenvalues and eigenvectors of their subdivision matrices determine the behaviors of the subdivision curves or surfaces. In a wavelet representation, a good predictor generally leads to smaller wavelet coefficients. Therefore, we use good subdivision filters as the prediction filters. For the updating step, we follow the idea in the example shown in section 7.2.1. We use wavelet redistribution to get a better approximation representation. A small error norm is important for a curve multiresolution representation. Integrating all these ideas, we can define our Wavelet Arc format—WArc. Table 7.2 gives the storage and transmission format

for a WArc. Compared to Table 7.1, it might appear that there is no reduction in data size. However, notice that the real reduction of data size comes from the smaller magnitudes of the wavelet coefficients, which will be compressed by different compression techniques such as those used in Chapter 4.

Total segments		*nseg*
Filter type		*filter*
Decomposition levels		N
Each segments	Initial points	V_0
	Wavelet coefficients (x,y,z)	$W_0, W_1, \ldots, W_{N-1}$

Table 7.2: WArc format

7.3 Implementation and Results

7.3.1 Wavelet Transform

We have implemented software based on the WArc format, in which the used wavelet transform is constructed using the lifting scheme. We used a simpler prediction step than the four-point scheme used in section 7.2.2, and the numerical results are quite acceptable. Equations (7.10), (7.11), and (7.12) show the three steps of the analysis transform in the lifting scheme. The synthesis transform can be obtained through the lifting scheme diagram. This wavelet transform is simple and efficient. It is good for scalable distributed GIS services.

$$\text{Partition} \quad \bar{c}_i^j = c_{2i}^{j+1} \ , \ \bar{d}_i^j = c_{2i+1}^{j+1} \ , \tag{7.10}$$

$$\text{Prediction} \quad d_i^j = \bar{d}_i^j - \tfrac{1}{2}(\bar{c}_i^j + \bar{c}_{i+1}^j) \ , \tag{7.11}$$

$$\text{Updating} \quad c_i^j = \bar{c}_i^j + \tfrac{1}{4}(d_i^j + d_{i-1}^j) \ . \tag{7.12}$$

In above equations, the given original data are a point sequence $\{c_i^N\}$, the finest data set. j is from $N-1$ to 0. Resolution 0 is the coarsest level representation. Notice that the above wavelet filter really gives the binlet 5/3 filter, which is stated

in equation (7.13).

$$\text{Effective filters} \begin{cases} d_i^j = -\frac{1}{2}c_{2i}^{j+1} + c_{2i+1}^{j+1} - \frac{1}{2}c_{2i}^{j+1} \\ c_i^j = -\frac{1}{8}c_{2i-2}^{j+1} + \frac{1}{4}c_{2i-1}^{j+1} + \frac{3}{4}c_{2i}^{j+1} + \frac{1}{4}c_{2i+1}^{j+1} - \frac{1}{8}c_{2i+2}^{j+1} \end{cases} . \tag{7.13}$$

A natural question is why a second generation wavelet gives the same filter as a first generation wavelet. The reason is that the connectivity information for a curve point sequence—the indices of the points—has provided a regular setting for the data. Because the coordinates of the given points are three independent sequences, the indices of the points become a natural regular setting for these three sequences. Therefore, an arbitrary point sequence of a curve can be decomposed by all first generation wavelets. This is the beauty of the lifting scheme, which connects first generation wavelets and second generation wavelets. However, in the next chapter, when we discuss a general triangular network, this will not happen since the indices of the points of a triangular patch cannot provide the connectivity information anymore. Only second generation wavelets can be used in that case.

7.3.2 Boundary Conditions

Similar to processing DEMs, boundary points need to be specially processed because the boundary points do not have some of the neighbors needed for prediction and updating. Generally, there are four approaches to deal with boundary conditions. They are constant padding, periodic extensions, mirror extensions, and extrapolating. In our implementation, a mirror extension is used to process the boundary conditions because it does not need extra memory to store data and it is easy to implement. The results are acceptable, as shown in the following numerical experiment.

7.3.3 Numerical Experiment Results

We have chosen one part of the Cape Cod (Massachusetts) coastline as numerical experiment data. The original Arc has 11 segments, totalling 2859 points. We decompose them into five resolutions. The results are shown in the figures below.

Figure 7-3 shows the original data. We decompose the data into five resolutions. Figure 7-4~Figure 7-7 show the results of the reconstructed coastline with 1, 2, 3, and 4 levels of wavelet coefficients. Notice that curve in Figure 7-7 is the same as the original data due to the perfect reconstruction. Finally, a comparison plot is given in Figure 7-8. It shows that using one half of the original data (i.e. a 3-level reconstruction) is enough for a general view. This is also useful in geographic data reduction and map generalization. Since curves are not our research major focus, we leave further experiments for a commercial implementation.

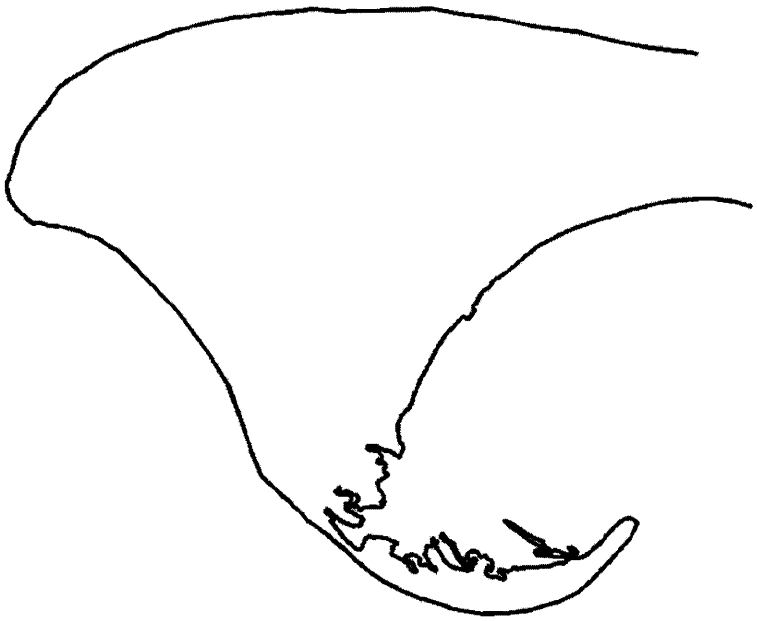


Figure 7-3: Original data

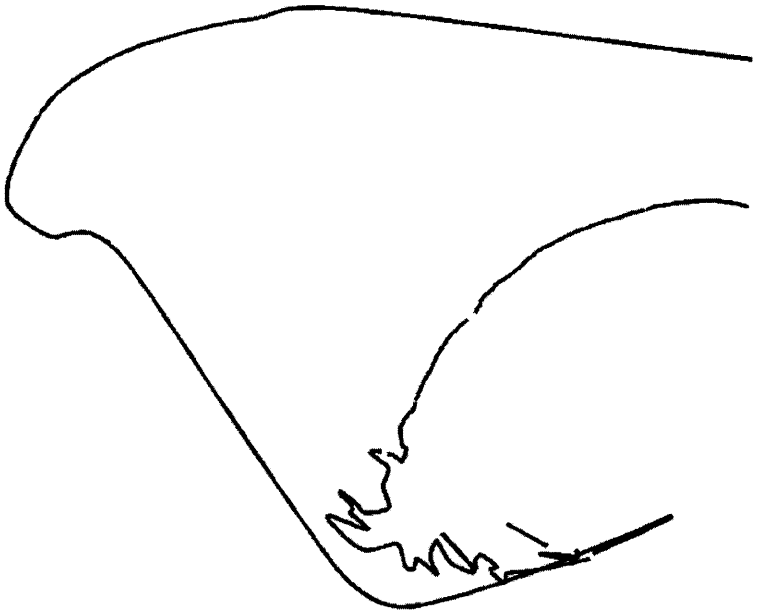


Figure 7-4: WArc representation with 1 level of wavelet coefficients

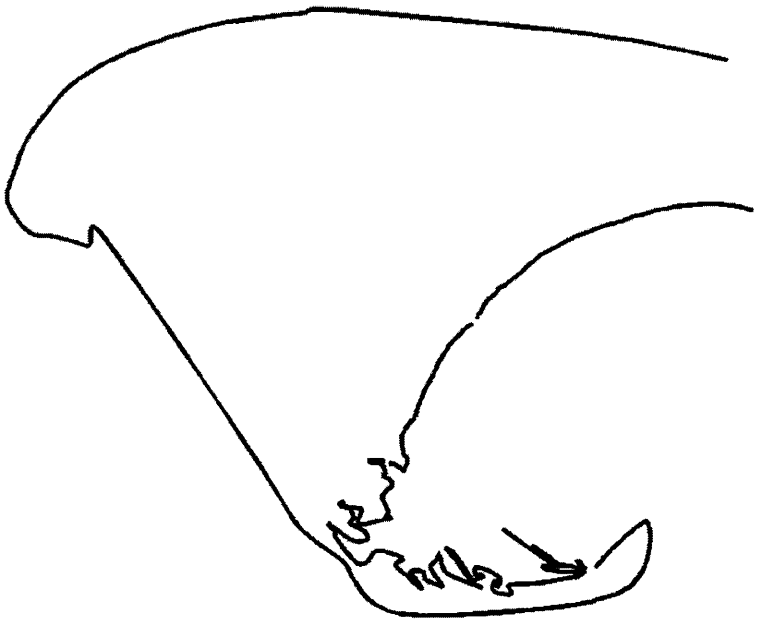


Figure 7-5: WArc representation with 2 levels of wavelet coefficients

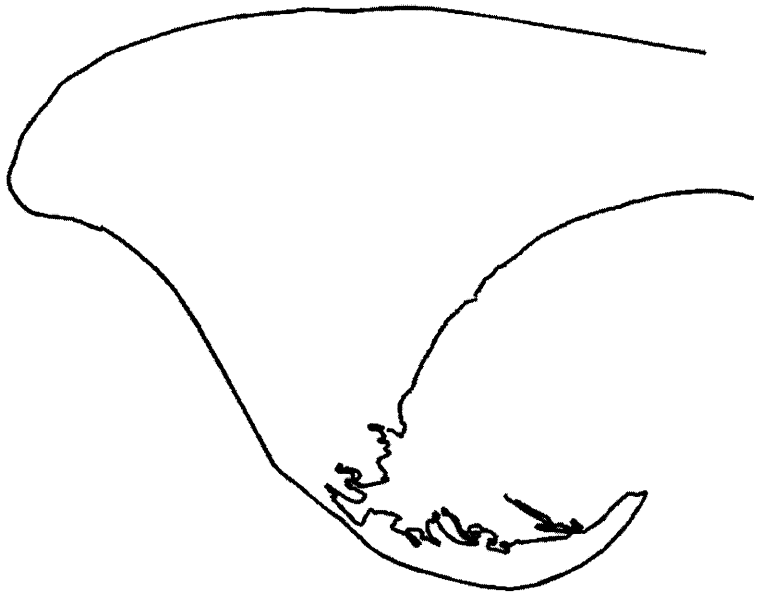


Figure 7-6: WArc representation with 3 levels of wavelet coefficients

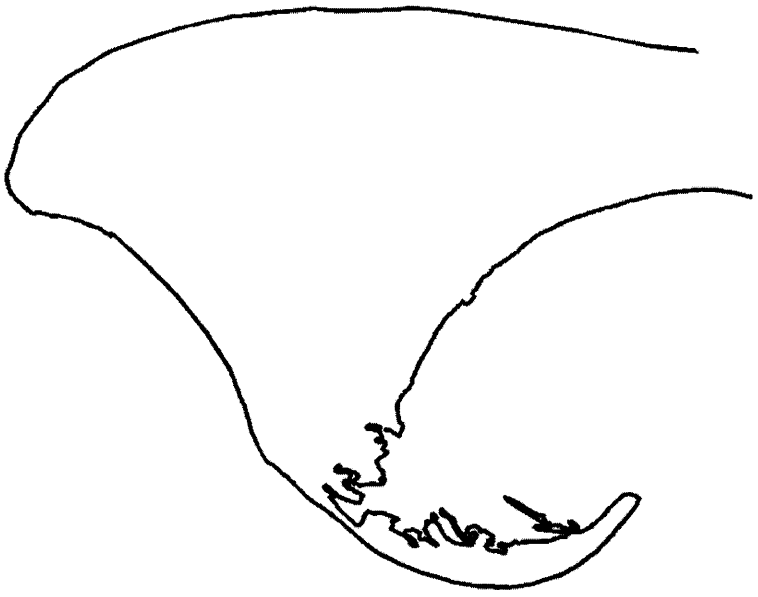


Figure 7-7: WArc representation with 4 levels of wavelet coefficients

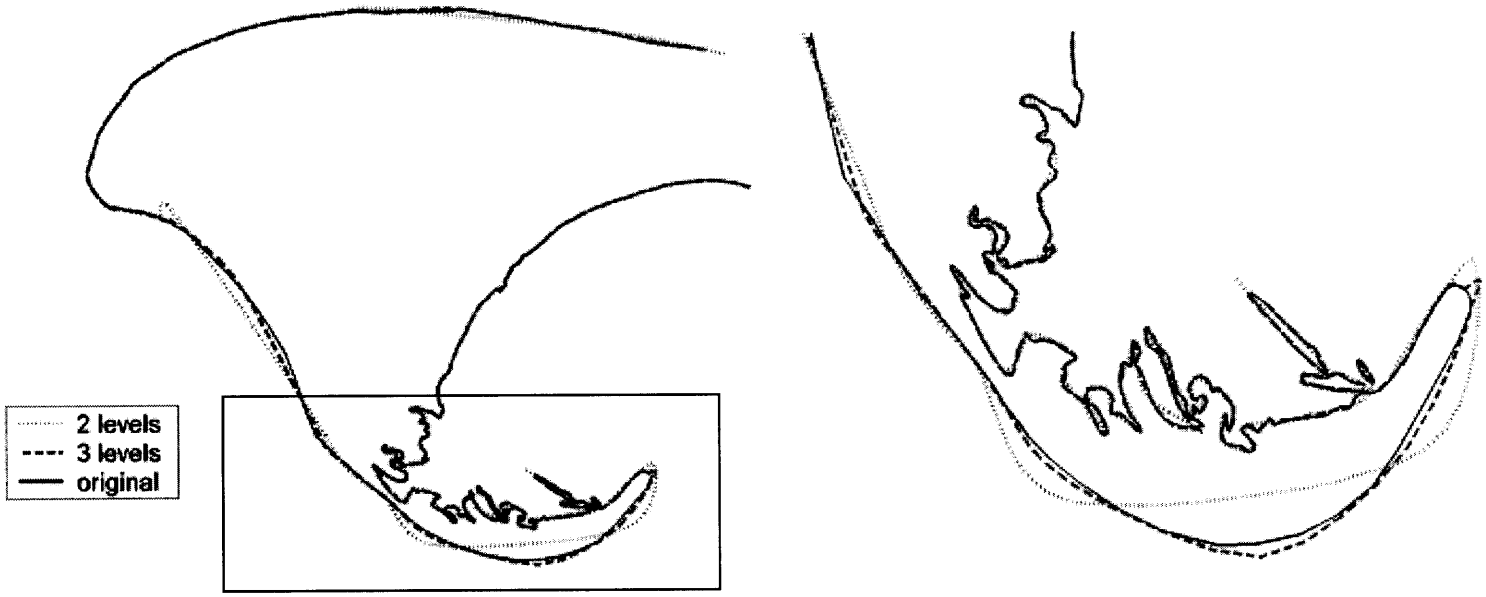


Figure 7-8: Comparison of different resolutions

Chapter 8

Wavelet Triangulated Irregular Networks (WTIN)

Chapter 7 has introduced a direct application of second generation wavelets to processing curves in GIS. More complex data in a GIS are surfaces. Chapter 5 has constructed one type of multiresolution data format for surfaces—WDEM; however, it is based on a regular setting, i.e. a uniform grid. Most GIS applications prefer to use irregular setting data since most data cannot be obtained for uniform grids, and DEMs are generally not good for computing the slopes and aspects of geographic features. In the GIS domain, many applications are based on Triangulated Irregular Networks (TINs), which have nonuniformly distributed points. Therefore, a new multiresolution format for TINs needs to be developed for GIS applications. This chapter firstly discusses the TIN format, and then constructs second generation wavelets for height fields. After that, a new data format—Wavelet Triangulated Irregular Network (WTIN)—is defined. And finally, numerical experiments are given to demonstrate the efficiency and potential of the WTIN format. An algorithm converting DEMs to WTINs is also given for practical usage.

8.1 Triangulated Irregular Networks (TINs)

In order to represent a terrain surface, a connected network typically needs to be built for non-uniformly spaced data. Triangulated networks are popular for expressing surface geometry because they typically contain fewer polygons then cell-based models, which results in faster rendering. Triangulated Irregular Networks (TINs) represent a surface as contiguous non-overlapping triangular faces. In this format, vertex coordinates and connectivity information are stored. They are organized as in Table 8.1. In a real implementation, some extra data, such as neighboring triangles, are stored as well; however, all those data can be derived from the basic data stored in Table 8.1. The extra storage is only for improving computational efficiency.

Total vertices	nv
Total faces	nf
Vertex coordinates	x_1, y_1, z_1 x_2, y_2, z_2 ...
Vertex connectivities	face 1: v_1^1, v_2^1, v_3^1 face 2: v_1^2, v_2^2, v_3^2 ...

Table 8.1: TIN format

Compared to DEMs, TINs have several advantages:

1. A TIN has variable vertex densities for different regions, more vertices on the regions with sharper geographic changes;

2. Precise locations for streams, ridges, and peaks can be recorded in TINs, while DEMs record features at points on cell-based grids;

3. Slope, aspect and volume calculations are more accurate based on TINs than based on DEMs;

4. Generally surfaces in TINs have less polygons than DEMs, therefore TINs are more efficient for rendering.

Based on these advantages, more GIS applications are based on TINs rather than on DEMs. However, the TIN format is not easy to represent in a multiresolution format. We have constructed the WDEM format based on first generation wavelets, which are built up from uniformly spaced points. TINs are triangles, the basic tool—the Fourier transform—in constructing first generation wavelets is not suitable for triangular data. Therefore, we must resort to a new tool—second generation wavelets— to construct wavelet transforms for TINs.

8.2 Wavelets for Height Fields

8.2.1 Height Field Characteristics

Three-dimensional terrain data are often treated as functions of two variables x and y. These surfaces are also called height fields. A basic characteristic of this type of data is that each point (x, y) has only one associated function value $f(x, y)$ (see Figure 8-1). This is the basic assumption in processing GIS surface data, whether they are in the DEM or the TIN format. There are situations where geographic surfaces have vertical slopes or folded surfaces, which need to be specially processed, either by different layering techniques or by volume models. This is not the case that we will discuss here. Therefore, one assumption of this research is that the geographical surfaces that we study are functions of two variables. This characteristic is a key factor that makes the proposed data representation more efficient than general three-dimensional geometric modeling representations.

Another assumption we need to make is that the coordinates of every point on the geographic surface of interest are known. Generally, we are given a series of points on the surface and we then use different interpolation or extrapolation methods to obtain the coordinates of the points in which we are interested. This process has been well studied. Therefore, we assume that every vertex can be obtained with sufficient accuracy.

A TIN is not suitable for direct use in scalable distributed GIS services because

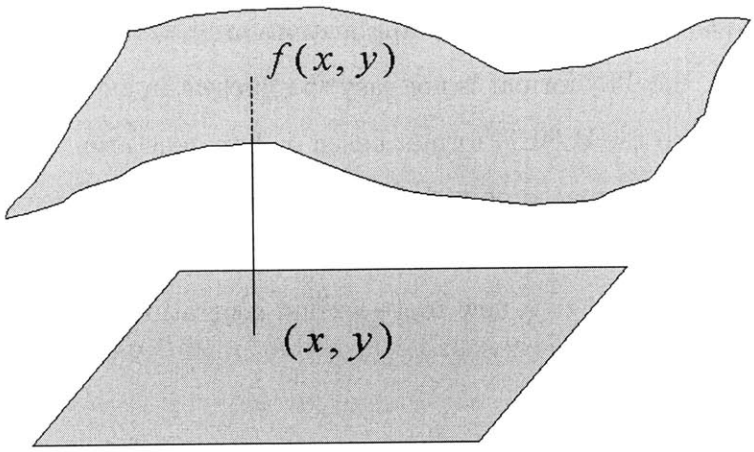
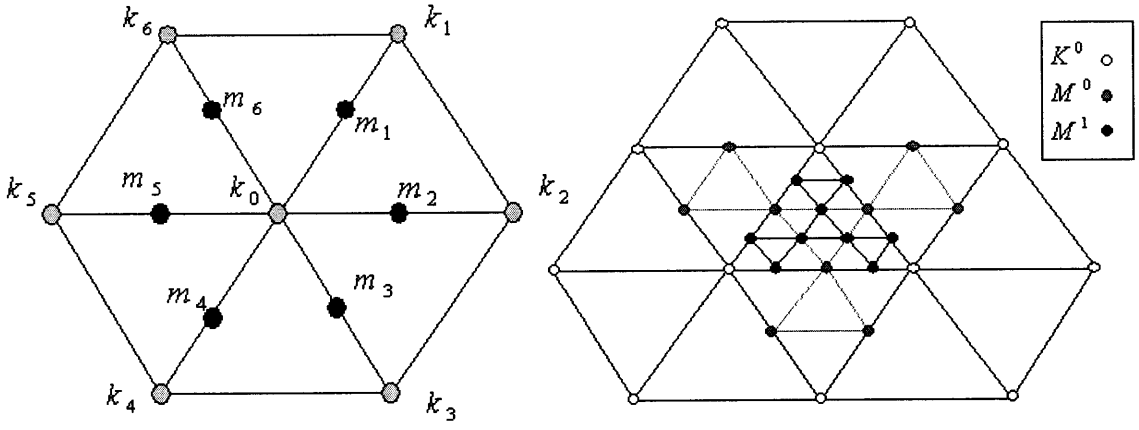


Figure 8-1: Height fields—functions of two coordinates

it does not have a natural multiresolution structure in its most general form. It needs extra memory to store the complex connectivity information. However, the cell structure in a DEM can easily tell us the connectivity information for a DEM. Furthermore, for a vertex in a DEM, only one value needs to be stored because the x and y coordinates can be determined from the cell structure (in this chapter, we always let x and y refer to the coordinate variables on the reference plane, such as in Figure 8-1). For a vertex in a TIN, three coordinates need to be remembered in the system since there is no general structure that can be used to determine the x and y coordinates. Therefore, if the vertex density is fixed, a DEM may have less data than a TIN because a DEM does not store the connectivity information. Fortunately, a TIN typically has a much smaller vertex density than a DEM in order to represent a surface. Some researchers have developed quaternary TINs and ternary TINs [16], which use hierarchical structures to store the connectivity information. These structures can be used to determine the x and y coordinates for some points as well. However, these schemes are based on simple linear subdivisions and heuristic rules. They are not suitable for visualization in general. Therefore, a new multiresolution format needs to be developed in order to obtain a better multiresolution representation for scalable distributed GIS services.

8.2.2 Structured Partition

From the introduction in Chapter 6 we know that there are three important steps to construct a second generation wavelet: a structured partition, good prediction, and good updating. A structured partition is the first step. It is the key component to build the hierarchical structure. For a fully unstructured triangular network, extra storage is always needed for the connectivity information; however, if we can build a structured partition, then all of the connectivity information is no longer needed. Figure 8-2 shows the structured partition that we used. It is conceptually a quaternary triangulation process. However, we need to notice that this figure only shows the topological information. Points belonging to different partitions are not necessarily along straight lines or at geometric mid-point positions. This figure presents only the topology. The symbols used in this figure are consistent with the symbols that we used in Chapter 6. In Figure 8-2(a), one partition process is explained. The original data are partitioned into two sets $\{k_i\}$ and $\{m_i\}$. Because they are in the same resolution, the resolution numbers are omitted. Figure 8-2(b) shows several levels of partition. $\mathcal{K}^j$ and $\mathcal{M}^j$ represent a partition of $\mathcal{K}^{j+1}$.



(a) (b)

Figure 8-2: Structured Partitions

Since the original data will not always have this hierarchical partition, a resampling process needs to be used to obtain the data at these points. This is based on the assumption that we can obtain the coordinates of every point on the geographic

surface of interest. This process will be further discussed in section 8.2.4. Here, we assume that the original data can be partitioned in this way.

8.2.3 Modified Butterfly Scheme

After the original data are partitioned into different sets, the correlation between these sets can be explored by a prediction step. Here, we only discuss two channel wavelet transforms; therefore, a prediction filter is applied on the first partition to predict the second partition.

From Chapter 6, we know that the construction of second generation wavelets has a close relationship with subdivision. A general subdivision scheme can be used as the prediction filter in the lifting scheme. Strictly speaking, a subdivision process has two steps. One is a splitting step; the other one is an evaluation step. The splitting step is more like a partition step, in which a triangle is divided into several sub-triangles. The evaluation step is to calculate the point values (point coordinates for a geometry) after the splitting step. Therefore, a subdivision scheme itself can be categorized into the interpolation type and the approximation type. In an interpolation subdivision scheme, once a point value is calculated, it will not change in future subdivision steps. In an approximation subdivision, previous point values will change in future subdivision steps. An interpolation subdivision is more attractive than an approximation one because we do not need to update the wavelet coefficients, which are already obtained, in further subdivision steps. This makes the computation faster.

For triangular networks, a good interpolation subdivision scheme is the Butterfly scheme as shown in Chapter 6. This scheme is based on quaternary triangulation and can achieve a C^1 continuous surface with a few exceptions. From Figure 8-2, it can be seen that the structured partitioning operation is connected to a quaternary subdivision process. One characteristic of such a triangulation is that an internal vertex always has 6 edges connecting to it. We define the term *valence* as the number of edges connecting to a vertex. Thus, a vertex with a valence of 6 is a regular vertex. Otherwise, the vertex is called an extraordinary vertex. Notice that this definition applies to internal vertices. For a vertex on the boundary, a valence of 4 is a regular

case, otherwise it is an extraordinary case. In order to better understand the Butterfly scheme, we can derive the butterfly scheme using a simple approach, which is not a proof though. In Figure 8-3, we map a general triangular network (plot (c)) to a standardized triangular network (plot (a) or (b)). This mapping involves following operations:

- select two parameters s and t;
- construct a two dimensional polynomial with $\{1, s, t, s^2, t^2, st, s^2t, st^2\}$ terms;
- use eight neighboring vertices to solve the polynomial coefficients.

Following the same procedure as for deriving the four-point scheme in section 7.2.1, one can obtain the subdivision scheme shown in Figure 6-5. For convenience, this figure is reproduced in Figure 8-3(c). The results from the two cases shown in Figure 8-3(a) and Figure 8-3(b) are the same. Vertex A is the evaluation reference point.

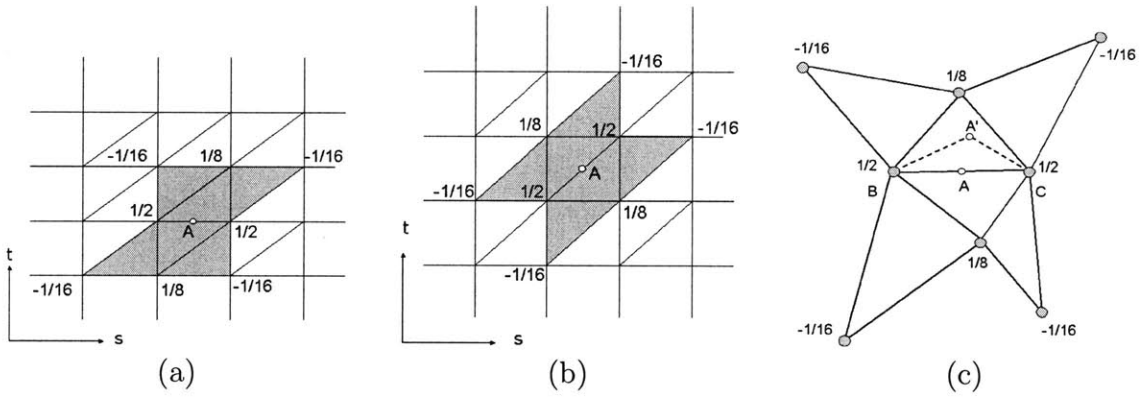


Figure 8-3: Constructing the Butterfly scheme

Figure 8-4 schematically shows the result of one subdivision step using the Butterfly scheme. Notice that we have not plotted the neighboring triangles.

The original Butterfly scheme can achieve C^1 continuity at internal regular vertices, but not at internal extraordinary vertices. Figure 8-5(a) shows an internal extraordinary vertex case where vertex Q has a valence of 5. Zorin *et al.* [53] dealt with extraordinary vertices in the Butterfly scheme by proposing a modified Butterfly scheme. Figure 8-5(b) illustrates the filter coefficients for predicting an internal vertex P', one of whose direct parent vertices, Q, is an internal extraordinary vertex. In

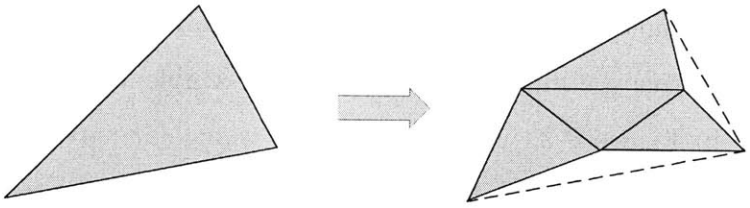


Figure 8-4: One step subdivision

the graph, $\{s_j\}$ and q are the predictor's coefficients for the corresponding reference point, P. These coefficients are defined in equation (8.1). P is the reference point for P', which is the result of applying the modified Butterfly scheme. In equation (8.1), k is the valence of the extraordinary point Q, which corresponds to the coefficient q. Notice that the points where $\{s_i\}$ apply need to be internal regular vertices.

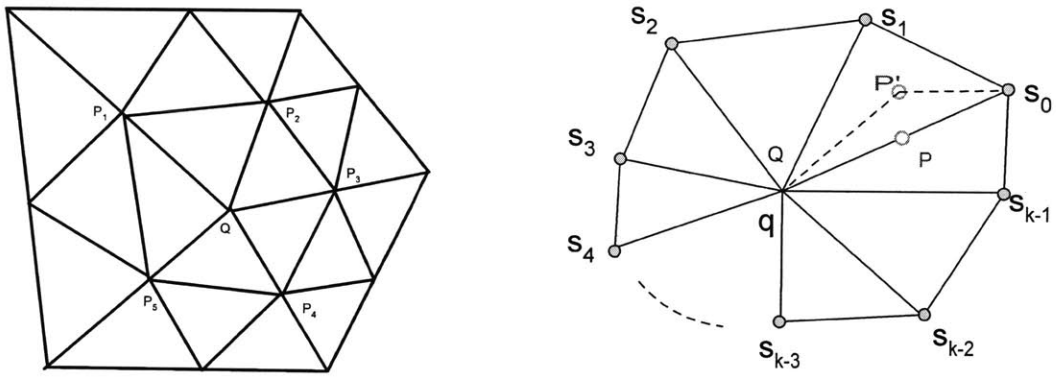


Figure 8-5: Modified Butterfly scheme

$$\begin{cases} s_0 = \frac{5}{12}, \quad s_1 = s_2 = -\frac{1}{12}, & k = 3 \\ s_0 = \frac{3}{8}, \quad s_2 = -\frac{1}{8}, \quad s_1 = s_3 = 0, & k = 4 \\ s_j = \frac{1}{k}(\frac{1}{4} + \cos(\frac{2\pi j}{k}) + \frac{1}{2}\cos(\frac{4\pi j}{k})), \ j = 0, \cdots, k-1, & k \geq 5 \\ q = 1 - \sum_{j=0}^{k-1} s_j \end{cases} . \tag{8.1}$$

Each time a new internal vertex is added, an appropriate filter from Figure 8-3(c) or Figure 8-5(b) will be applied based on the neighboring configuration. For the case

where the two direct parents of a new vertex are two internal extraordinary vertices, a simple average will be used on the results that are obtained from plugging each parent in Figure 8-5(b).

For boundary cases, Zorin [55] proposed several subdivision schemes for different cases. Figure 8-6 shows these cases and the corresponding subdivision schemes.

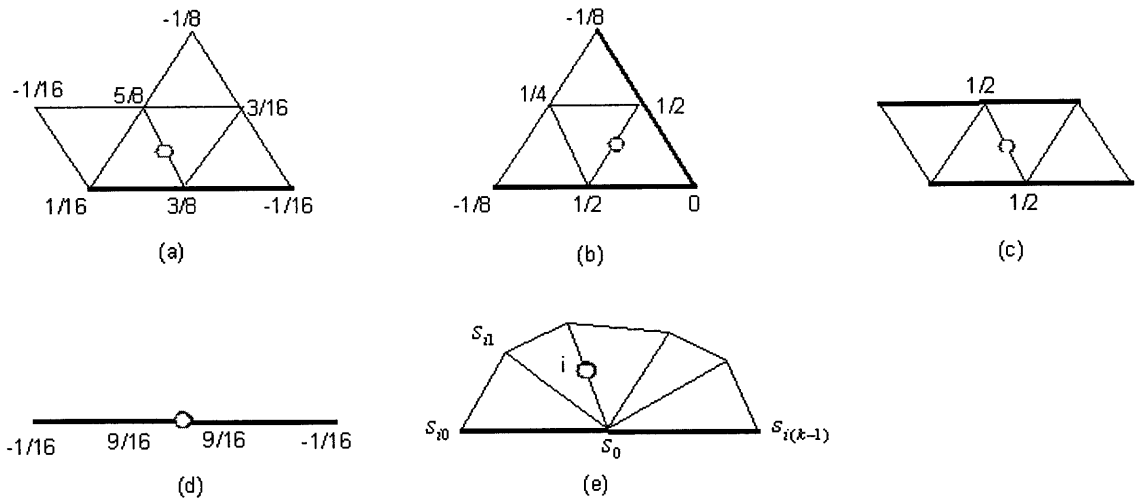


Figure 8-6: Boundary cases for the modified Butterfly scheme (Thicker edges are boundaries; circles denote points that are currently being computed)

The coefficients in Figure 8-6(e) are calculated in equation (8.2), where k is the valence of the boundary point to which s_0 applies.

$$\begin{cases} \theta_k = \dfrac{\pi}{k-1} \\ s_0 = 1 - \dfrac{1}{k-1}\dfrac{\sin\theta_k \sin(i\theta_k)}{(1-\cos\theta_k)} \\ s_{i0} = -s_{i(k-1)} = \dfrac{1}{4}\cos(i\theta_k) - \dfrac{1}{4(k-1)}\dfrac{\sin(2\theta_k)\sin(2i\theta_k)}{\cos\theta_k - \cos(2\theta_k)} \\ s_{ij} = \dfrac{1}{k-1}(\sin(i\theta_k)\sin(j\theta_k) + \dfrac{1}{2}\sin(2i\theta_k)\sin(2j\theta_k),\ j = 1, \cdots, k-2 \end{cases} \quad (8.2)$$

8.2.4 Interpolation Wavelet Filters for Height Fields

For a general three-dimensional object, every point has three coordinates x, y, and z. Therefore in a general three-dimensional wavelet transform, every point has three wavelet coefficients. However, a terrain surface, f, is a function of two variables, x

and y. The value $f(x, y)$ represents the true height value measured in the z direction at a projection point (x, y). If given the coarsest level data, one can run a subdivision process on them and obtain a subdivision surface, which is different from the true surface $f(x, y)$; however, every subdivision point $A(x_0, y_0, z_0)$ corresponds to a point $A'(x_0, y_0, f(x_0, y_0))$ on the true surface. Note that although points A' and A have different height values, one is $f(x_0, y_0)$ and the other is z_0, they have the same x and y coordinates. We can run a subdivision algorithm only on the x and y coordinates to obtain all the projections (x_i, y_i) of the real subdivision points (x_i, y_i, z_i) on the x–y plane. If we resample the true height surface on those points (x_i, y_i) and use them to express the terrain surface, then the wavelet coefficients for the x and y directions will be zero. We do not need to store and transmit the wavelet coefficients on the x and y directions. The only wavelet coefficient that needs to be stored for a point is the wavelet coefficient in the z direction, $f(x_i, y_i) - z_i$. This leads to a more efficient representation than those used in general three-dimensional object modeling techniques. Figure 8-7 shows the relationship between a projection point $(\hat{x}, \hat{y})$ in the x–y plane (really $(\hat{x}, \hat{y}, 0)$ in 3D), the corresponding subdivision point $(\hat{x}, \hat{y}, \hat{z})$ and the true surface point $(\hat{x}, \hat{y}, f(\hat{x}, \hat{y}))$.

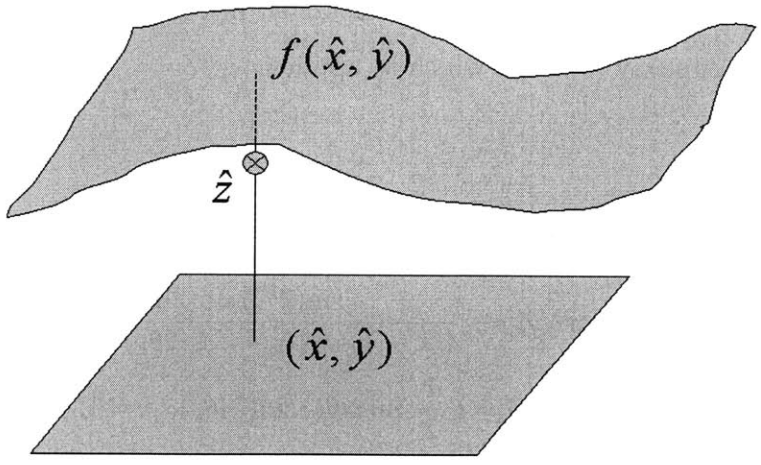


Figure 8-7: Interpolation wavelet transform for height fields

Before the wavelet transform can be applied, there is another important step, which is to determine the coarsest resolution configuration (also called the initial configuration). The initial configuration generally can be very simple, such as the

two triangles defined by the four corners of a rectangular area of interest, or some other simple triangular network defined by important points. A good strategy for determining the initial configuration is to look for extrema in the surface and connect them together. After determining the initial configuration, the height values at corresponding subdivision points need to be obtained from the raw geometry data. GIS terrain data come from various sources, such as satellite images, aerial photos, geographic surveys, and interpolation by technicians or computers, etc. A height value at a specific point can always be obtained with reasonable accuracy by some interpolation or extrapolation method provided the raw data is of a sufficiently high resolution. The inverse distance weighted interpolator and a spline interpolator are common tools. The following step, then, is to apply a wavelet transform on the data. Since many of the wavelet coefficients output from the analysis step are very small compared to the original height data, compression schemes find their place in processing the wavelet coefficients. The wavelet coefficients can be quantized to a desired error tolerance and then compressed by traditional compression schemes such as those used in image compression (see Chapter 4). Finally, on the other side, the required data can be synthesized from those wavelet coefficients and the coarsest level data. The whole process is shown in Figure 8-8.

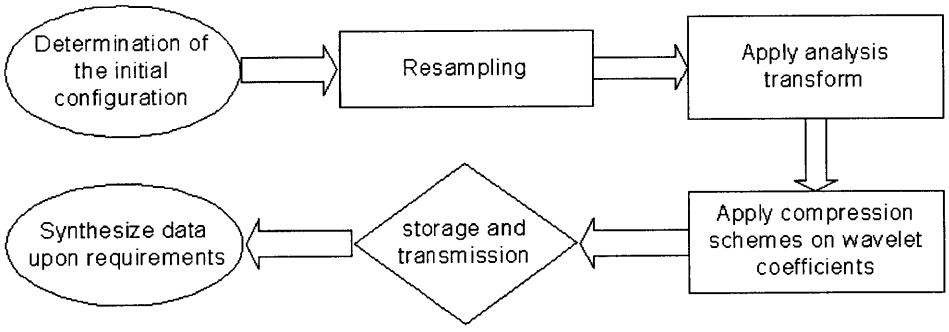


Figure 8-8: Procedure for processing terrain data by second generation wavelets

The wavelet transforms used in this process are given below:

- The analysis transform:

$$
\left\{
\begin{aligned}
c_{\mathcal{K}^N} &= \begin{bmatrix} x_{\mathcal{K}^N} \\ y_{\mathcal{K}^N} \\ f(x_{\mathcal{K}^N}, y_{\mathcal{K}^N}) \end{bmatrix} \\
&\left\{
\begin{aligned}
\mathcal{K}^i &= \mathcal{K}^{i-1} \cup \mathcal{M}^{i-1} \\
\bar{c}_{\mathcal{K}^{i-1}} &= c_{\mathcal{K}^i}(\mathcal{K}^{i-1}) \\
\bar{d}_{\mathcal{M}^{i-1}} &= c_{\mathcal{K}^i}(\mathcal{M}^{i-1}) \\
d_{\mathcal{M}^{i-1}} &= \bar{d}_{\mathcal{M}^{i-1}} - P(\bar{c}_{\mathcal{K}^{i-1}}) \\
&= \begin{bmatrix} x_{\mathcal{M}^{i-1}} = \hat{x}_{\mathcal{M}^{i-1}} \\ y_{\mathcal{M}^{i-1}} = \hat{y}_{\mathcal{M}^{i-1}} \\ f(x_{\mathcal{M}^{i-1}}, y_{\mathcal{M}^{i-1}}) \end{bmatrix} - \begin{bmatrix} \hat{x}_{\mathcal{M}^{i-1}} \\ \hat{y}_{\mathcal{M}^{i-1}} \\ \hat{z}_{\mathcal{M}^{i-1}} \end{bmatrix} \qquad , i = N, N-1, \cdots, 1 \\
&= \begin{bmatrix} d_{\mathcal{M}^{i-1},x} = 0 \\ d_{\mathcal{M}^{i-1},y} = 0 \\ d_{\mathcal{M}^{i-1},z} = f(x_{\mathcal{M}^{i-1}}, y_{\mathcal{M}^{i-1}}) - \hat{z}_{\mathcal{M}^{i-1}} \end{bmatrix} \\
c_{\mathcal{K}^{i-1}} &= \bar{c}_{\mathcal{K}^{i-1}}
\end{aligned}
\right.
\end{aligned}
\right. \tag{8.3}
$$

- The synthesis transform:

$$
\left\{
\begin{aligned}
\bar{c}_{\mathcal{K}^i} &= c_{\mathcal{K}^i} \\
\bar{d}_{\mathcal{M}^i} &= d_{\mathcal{M}^i} + P(\bar{c}_{\mathcal{K}^i}) \\
&= \begin{bmatrix} 0 \\ 0 \\ d_{\mathcal{M}^i,z} \end{bmatrix} + \begin{bmatrix} \hat{x}_{\mathcal{M}^i} \\ \hat{y}_{\mathcal{M}^i} \\ \hat{z}_{\mathcal{M}^i} \end{bmatrix} \\
&= \begin{bmatrix} x_{\mathcal{M}^i} = \hat{x}_{\mathcal{M}^i} \\ y_{\mathcal{M}^i} = \hat{y}_{\mathcal{M}^i} \\ d_{\mathcal{M}^i,z} + \hat{z}_{\mathcal{M}^i} \end{bmatrix} \qquad , i = 0, 1, \cdots, N-1 \\
\mathcal{K}^{i+1} &= \mathcal{K}^i \cup \mathcal{M}^i \\
c_{\mathcal{K}^{i+1}}(\mathcal{K}^i) &= \bar{c}_{\mathcal{K}^i} \\
c_{\mathcal{K}^{i+1}}(\mathcal{M}^i) &= \bar{d}_{\mathcal{M}^i} \\
c_{\mathcal{K}^N} &= \begin{bmatrix} x_{\mathcal{K}^N} \\ y_{\mathcal{K}^N} \\ f(x_{\mathcal{K}^N}, y_{\mathcal{K}^N}) \end{bmatrix}
\end{aligned}
\right. \tag{8.4}
$$

Equation (8.3) is the analysis transform. It differs from the general analysis transform in Chapter 6 in that the wavelet coefficients in the x and y directions are always zeros, which reduces the storage space of the wavelet coefficients by 2/3. The updating term disappears because an interpolation filter is the goal. Equation (8.4) gives the synthesis transform, which simply reverses the process in equation (8.3)—a key advantage of the lifting scheme. The symbols $\bar{c}$ and $\bar{d}$ are intermediate symbols to represent partitioning results. $c_{\mathcal{K}^i}$ is partitioned into two components: $c_{\mathcal{K}^i}(\mathcal{K}^{i-1})$ and $c_{\mathcal{K}^i}(\mathcal{M}^{i-1})$, which belong to sets $\mathcal{K}^{i-1}$ and $\mathcal{M}^{i-1}$ respectively. Based on equation (8.3), the original data $c_{\mathcal{K}^N}$ is decomposed into $c_{\mathcal{K}^0}$, $d_{\mathcal{M}^0}$, $d_{\mathcal{M}^1}$, ..., $d_{\mathcal{M}^{N-1}}$. The synthesis transform reconstructs $c_{\mathcal{K}^0}$, $c_{\mathcal{K}^1}$, ..., $c_{\mathcal{K}^N}$, which yield a multiresolution representation of the original data.

The above transform provides a description of an interpolation wavelet transform, which does not include an updating filter such as in the general lifting scheme. This satisfies the interpolation requirement for some GIS applications. Figure 8-9 illustrates one step of the transform. Other GIS applications may not have such an interpolation requirement, then an updating filter can be added, however, currently there is no general theory about how to effectively design a good updater for triangular networks.

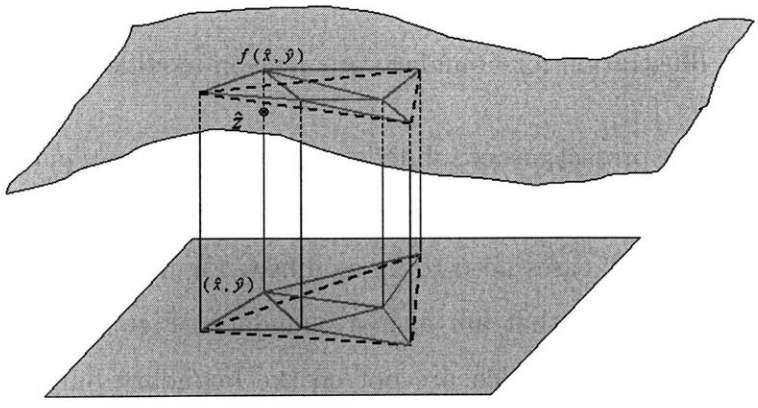


Figure 8-9: One step of the interpolation wavelet transform for height fields

8.3 Boundary Problems and Modifications

For terrain surfaces in the GIS domain, a rectangular boundary in the x–y plane is often used. Figure 8-6 has already shown the boundary cases for the modified Butterfly scheme. However, the filters may result in new subdivision points which are outside the boundary of the original GIS data. This is because when we run the subdivision algorithm on the x and y coordinates of the coarse resolution data, the new subdivision points are not geometrically bounded by the coarse resolution data. The reason for this effect is that the modified Butterfly scheme tries to smooth a local region near an extraordinary point, such as the point Q in Figure 8-5(b); while the nearby points have some tangential movement around this extraordinary point, which may cause these points move out of the boundary. This is shown in Figure 8-10. This problem does not appear in processing a general 3D object because there is no such a rectangular boundary constraint.

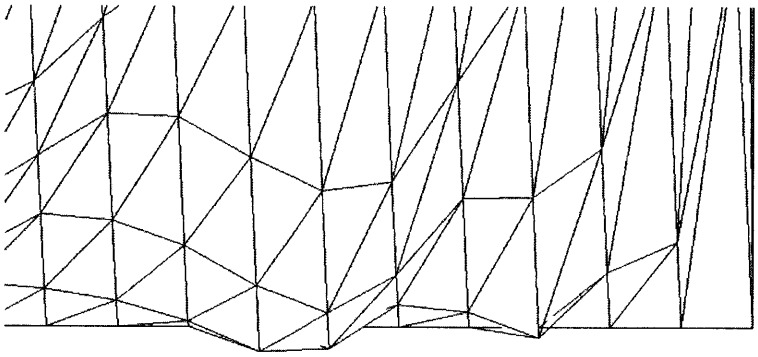


Figure 8-10: Illustration for boundary problems in modified Butterfly scheme

In our model, the projections of subdivision points on the x–y plane will be used to resample a terrain surface to obtain the most detailed resolution data (section 8.2.4). Therefore, the boundary cases need to be modified. Figure 8-11 shows all the filters used for processing points that are near a boundary. Figures 8-11(a)$\sim$(e) are for the new subdivision points which are not on the boundary but with at least one direct parent on the boundary. Other plots are for new subdivision points which reside on the boundary. The basic idea is to use a filter which ensures that the new subdivision points appear inside the current geometry. This generally results in a

filter with smaller support, i.e. fewer coefficients, such as in Figure 8-11(c). In this case, an average filter is used because it guarantees that the boundary condition is satisfied although the surface smoothness may be reduced. From a practical view, this reduction of smoothness near boundary does not harm the quality of the final surface as a whole. Figure 8-11(a) is derived from Figure 8-6(e) with $k = 4$. Figure 8-11(e) is the same as Figure 8-6(e). A corner type vertex is added because a corner vertex cannot be treated as a general boundary point for height fields. The neighboring vertices of a corner vertex should be processed separately because they do not lie on the same boundary curve; however, the neighboring vertices of a general boundary vertex can be processed using a curve approximation scheme, such as in Figures 8-11(f)~(i). A complete set of rules for dealing with internal and boundary points is given in Table 8.2. There are a total of 15 cases for adding a new internal point and 3 cases for adding a new boundary point. In order to process all these cases, 11 rules are proposed.

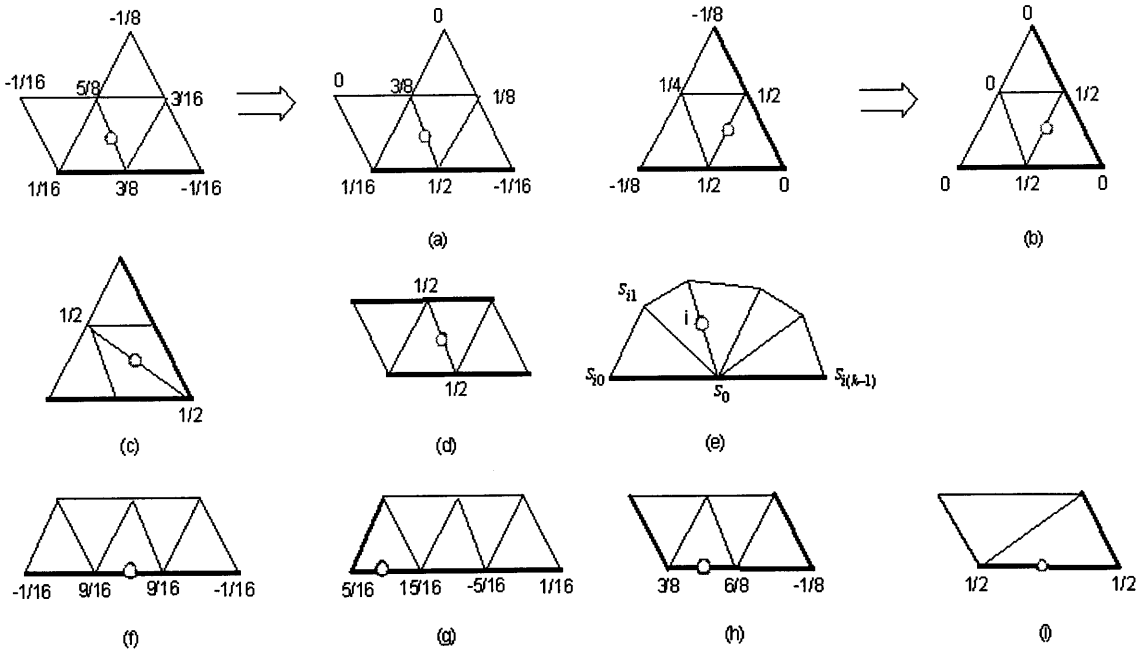


Figure 8-11: Illustrations for modified boundary cases (Thicker edges are boundaries; circles denotes points that are currently being computed)

Case		Rule	Notes
A new internal subdivision point	IR-IR	1	IR: internal regular point (valence is 6)
	IR-IE	2	IE: internal extraordinary point (valence is not 6)
	IR-BR	3	BR: boundary regular point (valence is 4)
	IR-BE	5	BE: boundary extraordinary point (valence is not 4)
	IR-C	4	C: corner point
	IE-IE	9	BD: intermediate point on a boundary (BR or BE)
	IE-BR	2	
	IE-BE	11	rule 1: Figure 8-3(c)
	IE-C	4	rule 2: Figure 8-5(b)
	BR-BR	4	rule 3: Figure 8-11(a)
	BR-BE	5	rule 4: Figure 8-11(b),(c),(d),(i)
	BR-C	4	rule 5: Figure 8-11(e)
	BE-BE	10	rule 6: Figure 8-11(f)
	BE-C	4	rule 7: Figure 8-11(g)
	C-C	4	rule 8: Figure 8-11(h)
A new boundary subdivision point	BD-BD	6	rule 9: apply Figure 8-3(c) on both parents and average the results
	BD-C	7 or 8	rule 10: apply Figure 8-11(e) on both parents and average the results
	C-C	4	rule 11: apply Figure 8-3(c) and Figure 8-11(e) on corresponding parents and average the results

Table 8.2: Complete boundary rules for processing GIS terrain data in a WTIN

8.4 Data Formats

Since only one wavelet coefficient needs to be stored for each vertex, the storage and transmission data format becomes simple and compact. However, the online computational structures are more complex than a WDEM.

8.4.1 Data Format for Storage and Transmission

Based on the above discussion, a new data format—Wavelet Triangulated Irregular Networks (WTIN)—is designed. The storage and the transmission of GIS terrain surface data in this format require only four types of information as shown in Table 8.3.

Control information	$N(V_0), N(F_0), N(W), T, L$	integer [5]
V_0	(x_0, y_0, z_0), ...	real $[N(V_0), 3]$
F_0	(v_1, v_2, v_3), ...	integer $[N(F_0), 3]$
W	$w_1, w_2, w_3, w_4, \ldots$	real $[N(W)]$

Table 8.3: Data format for storage and transmission of WTINs

The meanings of all variables in Table 8.3 are listed below:

- V_0 denotes the vertices in the initial configuration, which includes the three coordinates of all the initial vertices;
- F_0 denotes the faces in the initial configuration, which hold the labels of the three vertices;
- W denotes the wavelet coefficients for all subdivision points. For scalable distributed GIS services, W can be divided into several array objects, which correspond to different resolutions. Then the requested levels of wavelet coefficients can be transmitted separately as different objects;
- $N(\cdot)$ is the number of items in $(\cdot)$;
- T is the type of the wavelet transform used in processing the data;
- L is the total number of resolution levels included in the data.

Here, T allows for a future extension to use other wavelet filters. In this work, the interpolation wavelet filter described in section 8.2.4 is used. Notice that in this format, only one wavelet coefficient, in the z (height) direction, is stored for each subdivision point. The reason can be clearly seen in equation (8.3). Because many coefficients in W are very small compared to the original data, compression schemes can be used to make the total data size smaller. A detailed analysis is given in section 8.6.

8.4.2 Data Structures for Online Computation

Although the storage and transmission format of WTINs is very compact, more complex data structures need to be used for the analysis and synthesis transforms in order to retrieve the subdivision topology information. Figure 8-12 and Figure 8-13 show the proposed data structures to dynamically host the intermediate data.

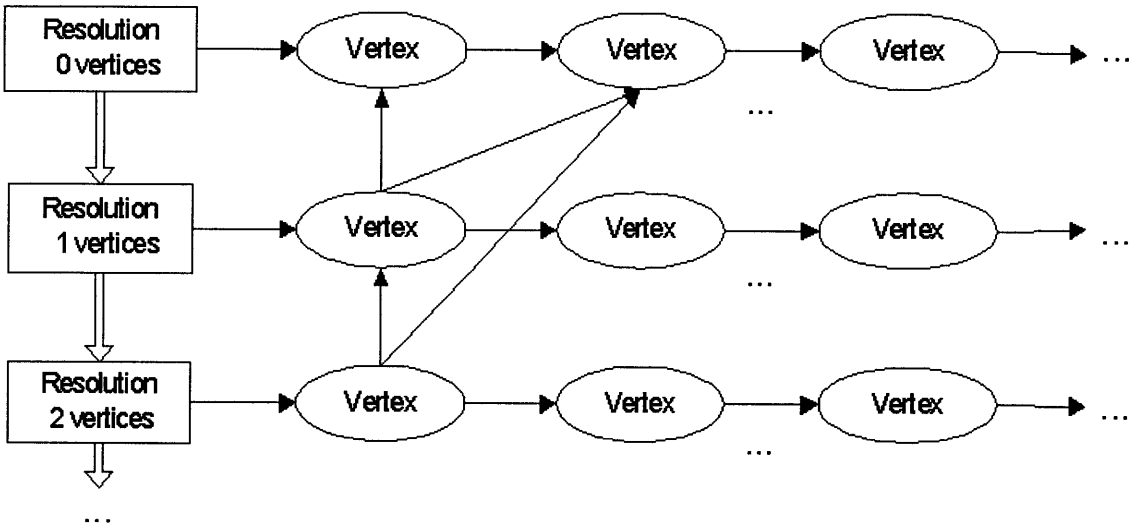


Figure 8-12: Multi-linked list for storing vertex information of WTINs

The vertex information is stored in a multi-linked list structure, and face information is stored in a quadtree array. For each vertex, two vertex labels (pointers) for the direct parents are needed besides the coordinates. For the vertices in the initial configuration, the direct parents are null. The vertex type is also stored in order to apply the correct rule in Table 8.2. For every face, three defining vertices,

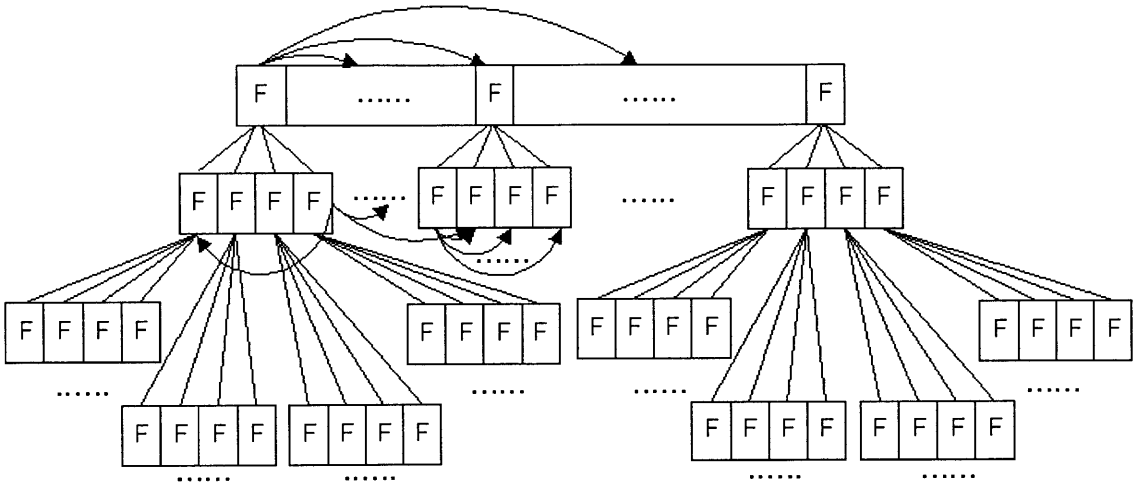


Figure 8-13: Quadtree-array for storing face information in WTINs

three neighboring faces (null if the neighbor is empty), and the resolution level are required. The neighboring face information permits the modified butterfly scheme to be implemented efficiently.

8.4.3 Class Definitions

Table 8.4 gives the Java class definitions for vertices and faces. All the required information in the computational process can be derived from these data. Essentially, the information in Table 8.3 will determine all the data through the wavelet transform. The intermediate data structures make the computation more efficient.

```
public class Vertex {
   private double x, y, z;
   private int type;
   private Vertex[] parents = new Vertex[2];
   //methods ...;
}
public class Face {
   private Vertex[] vertex = new Vertex[3];
   private int level;
   private Face[] nb = new Face[3];
   //methods ...
}
```

Table 8.4: Java class definitions for vertices and faces in WTINs

8.5 Determination of the Initial Configuration

In the previous sections we have discussed the construction of the WTIN format; however, the first step in the computational process—determination of the initial configuration—have not been discussed in depth. In reality, this is an important step in the whole computational process. All the subdivisions are based on this configuration. A basic principle in this step is to obtain the geometric features that we want to represent in the coarsest resolution, such as peaks of mountains. Therefore, this process may involve a human's decision. For a general terrain surface, if no predetermined features need to be preserved, we still want to keep those local extreme points, such as local peaks and valleys. Sometimes, we only want a few key points in the initial configuration. These cases can be programmed into an automatic process, which is the focus of this section.

Using a high resolution DEM as an example, we want to automatically abstract an initial configuration to satisfy several requirements. The requirements are the following:

- The four corners of the rectangular region of interest need to be in the initial configuration;
- The remaining points in the initial configuration need to be local feature points, such as peaks, valleys, or inflection points;
- Users can specify the approximate number of points in the initial configuration.

These requirements lead to a reasonably good representation of a terrain surface. The following algorithm was developed to determine the initial configuration automatically.

1. Set the target number of points in the initial configuration as a user input.
2. Go through all the DEM points to find the local extreme points. Local extreme points are defined as feature points, which are higher or lower than all other neighboring points. In a DEM, each point has eight neighboring points. So this step includes a large number of comparisons.

3. The four corners are always included in the initial configuration. Reduce the target number of points by four.

4. Divide-and-conquer:

   - The original rectangle is divided into four sub-rectangles.
   - The target number of points to be picked up in each sub-rectangle is proportional to the number of feature points in this sub-rectangle. Approximate the target number for a sub-rectangle to an integer.
   - If a sub-rectangle only has one target point assigned, choose the feature point closest to the center of the sub-rectangle.
   - Iteratively subdivide each sub-rectangle until all target points are assigned.

5. Construct the Delaunay triangular network on all the selected initial points.

This algorithm gives a Delaunay triangular network whose number of vertices is approximately specified in advance. All vertices are local extreme points. For the extreme case where the studied area is flat, we use four corners. From our experiments, this algorithm is an efficient way to obtain an acceptable initial configuration. Figure 8-14 demonstrates the basic idea and shows an example with real data, which will be used in section 8.6. The user input for the number of points in the initial configuration is ten. The final number of output points in the initial configuration is nine (including the four corners) due to the approximation in the algorithm, which satisfies the requirement.

8.6 Numerical Results

In order to verify the effectiveness of the proposed WTIN format, a software prototype for processing three-dimensional multiresolution GIS data has been developed. The software was developed using the Java programming language in order to plug into online GIS applications in the future. The software is designed using an object-oriented

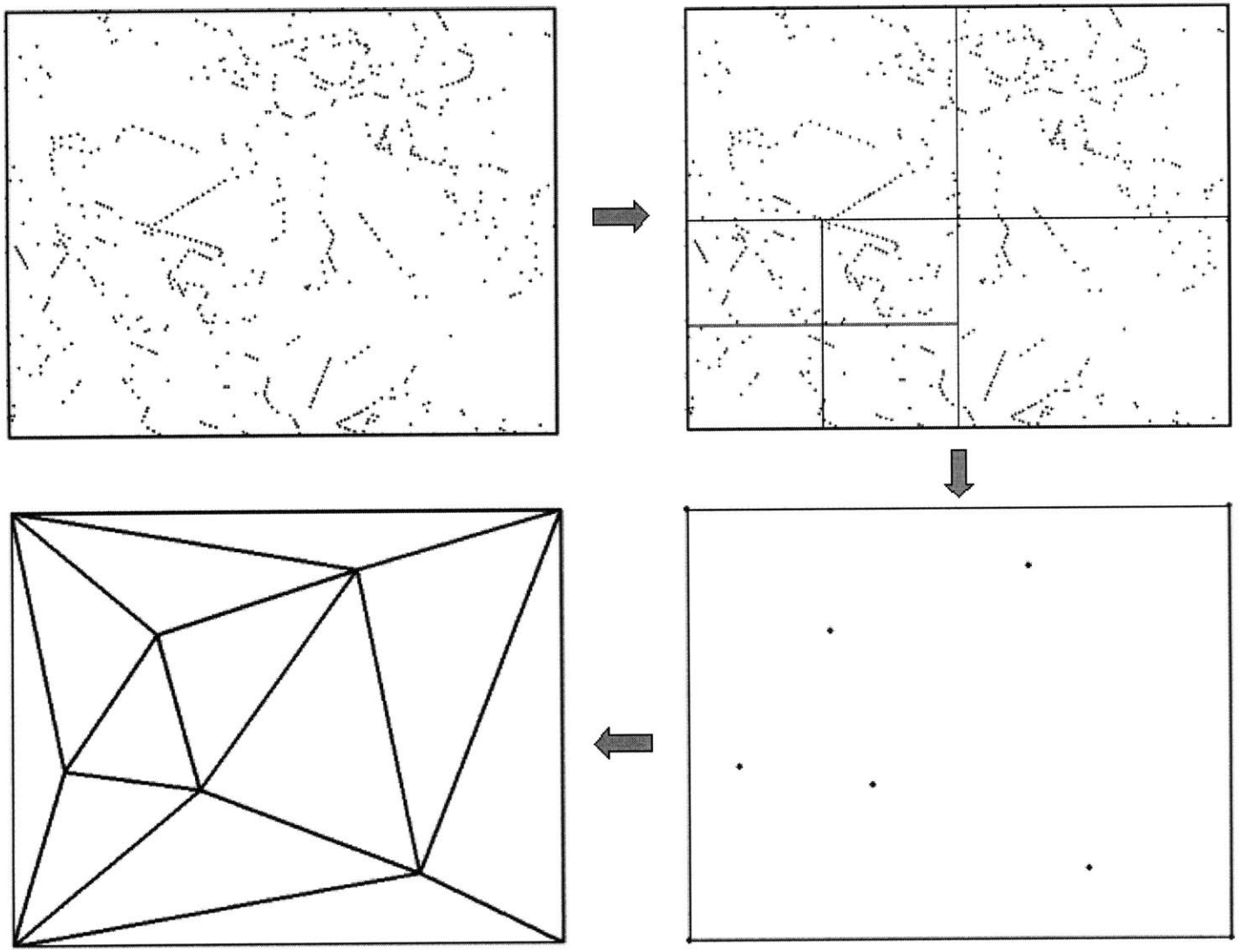


Figure 8-14: Constructing the initial configuration

model; the core classes that are provided are CompactWtin, Wtin, MbutterflyFilter, MultiLinkedlist, QuadTreeArray, Vertex and Face.

The original data source is a high resolution USGS DEM. The algorithm introduced in section 8.5 is used to determine the initial configuration, which has 9 vertices and the 12 corresponding Delaunay triangles. After that, a bilinear interpolation algorithm is used to obtain height values at all subdivision points. This gives acceptable accuracy as long as the original DEM grid is sufficiently dense. Figure 8-15 shows the original DEM data. The interpolation wavelet transform introduced in section 8.2.4 is used to process these data. Figure 8-16 shows the top view of the initial configuration. Based on this initial configuration, 7 resolutions are constructed during wavelet analysis.

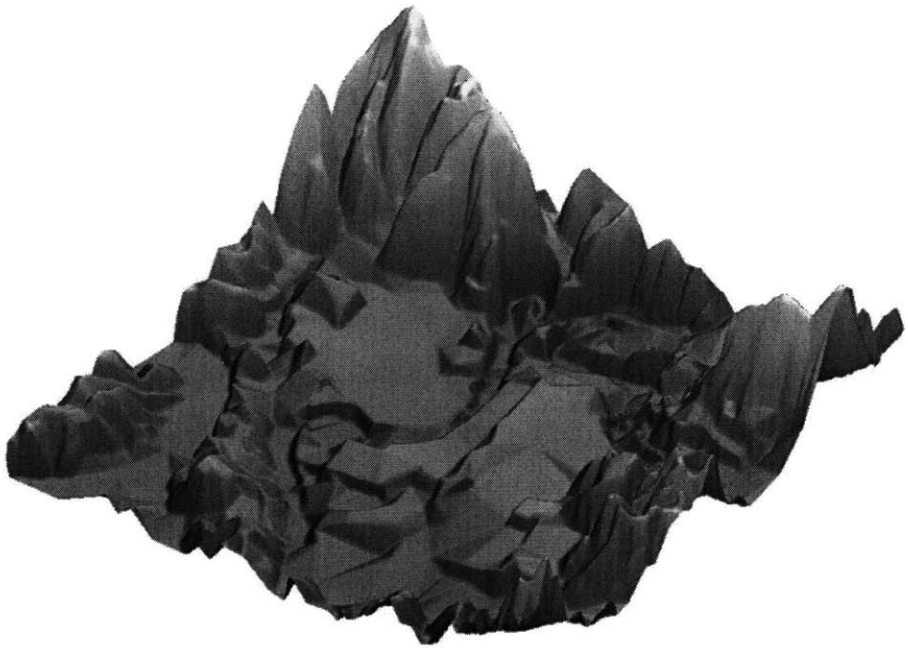


Figure 8-15: Original DEM

Figure 8-17 is the histogram and cumulative distribution function (CDF) of wavelet coefficients. From this plot, one can see that most coefficients fall within 5% of the height range (for this example, the height range is 103.5m and 96% of the wavelet coefficients fall in the 5% range). Therefore, compression schemes, such as Huffman

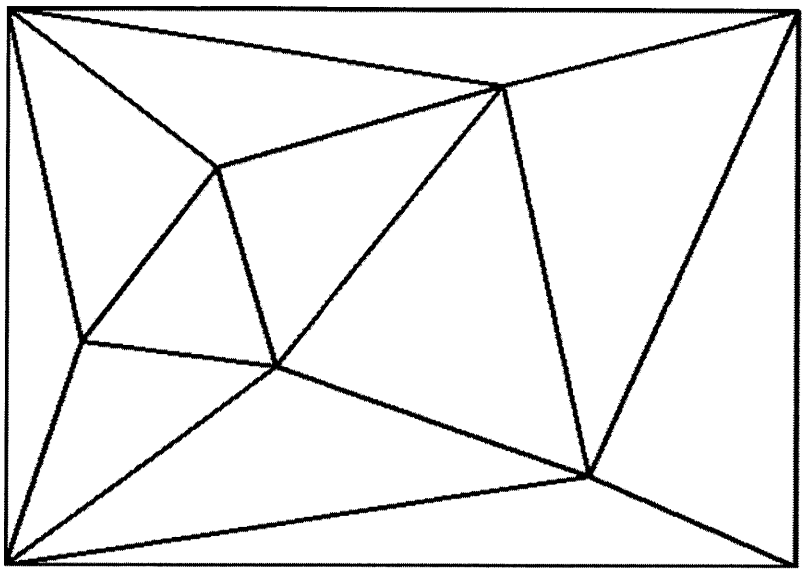


Figure 8-16: Top view of initial configuration

coding or Arithmetic coding, can be applied to these wavelet coefficients to reduce the storage data size. In our example, a simple thresholding operation is applied to the wavelet coefficients instead of the more complex compression techniques that were discussed in Chapter 4. The compressed result using 3% of the height range as the compression threshold is shown in Figure 8-18. This has a corresponding compression ratio of 11:1. A more carefully designed quantization technique can achieve better compression. In order to see the information content in each level of a WTIN, one can build a surface by using only a chosen subset of the wavelet coefficients consisting of the first several levels. To do this, one first uses the subset of wavelet coefficients to perform the synthesis transform, and then applies pure subdivision to the result from the synthesis transform. Figures 8-19, 8-20, 8-21, 8-22 give four representations using the first 2, 3, 4, and 5 levels of wavelet coefficients respectively to build the terrain. The percentage of the total wavelet coefficients retained is given under each figure.

A quantitative analysis of the quality of reconstructed surfaces is given based on

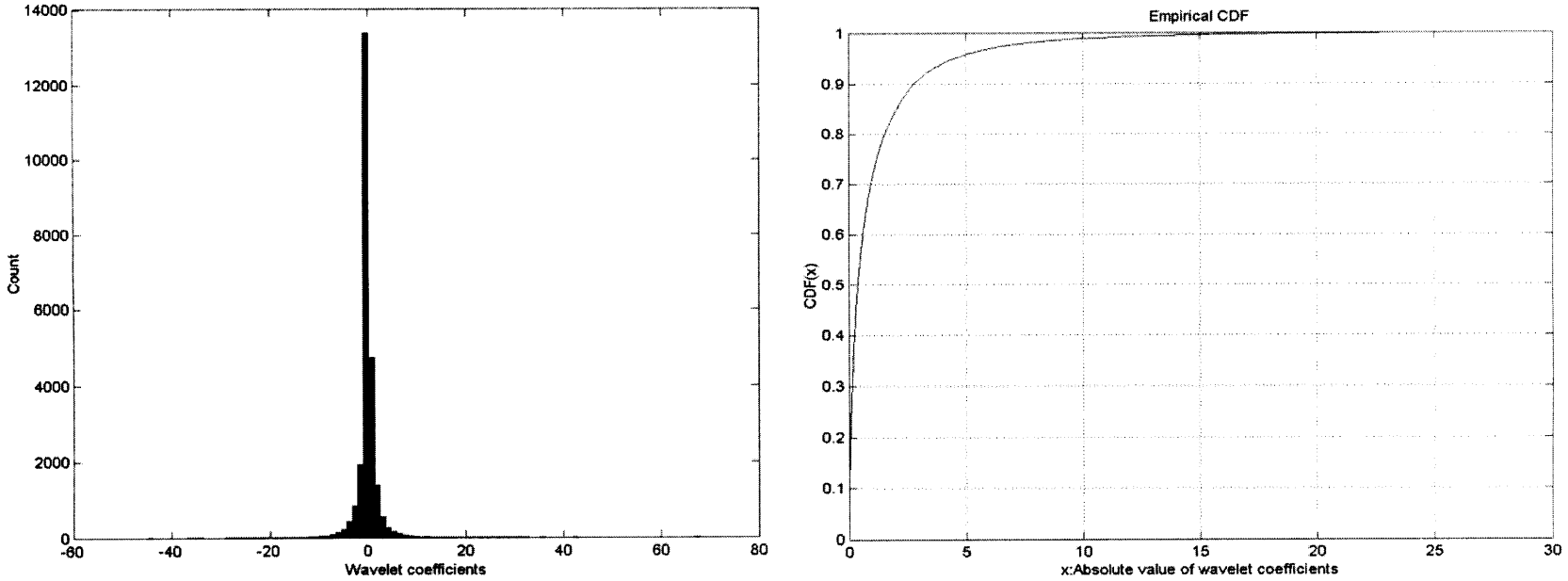


Figure 8-17: Distribution of magnitudes of wavelet coefficients

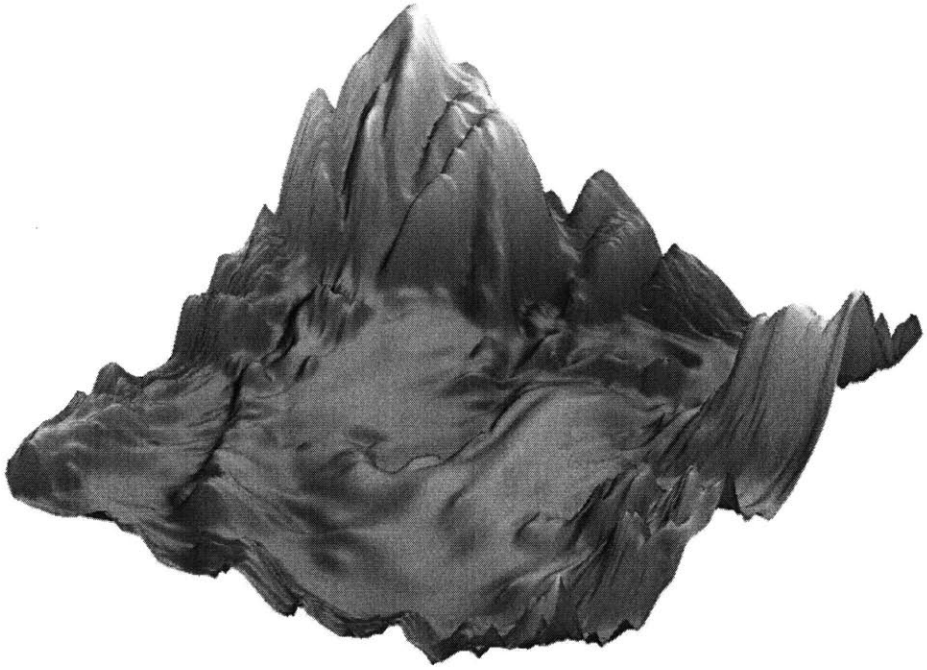
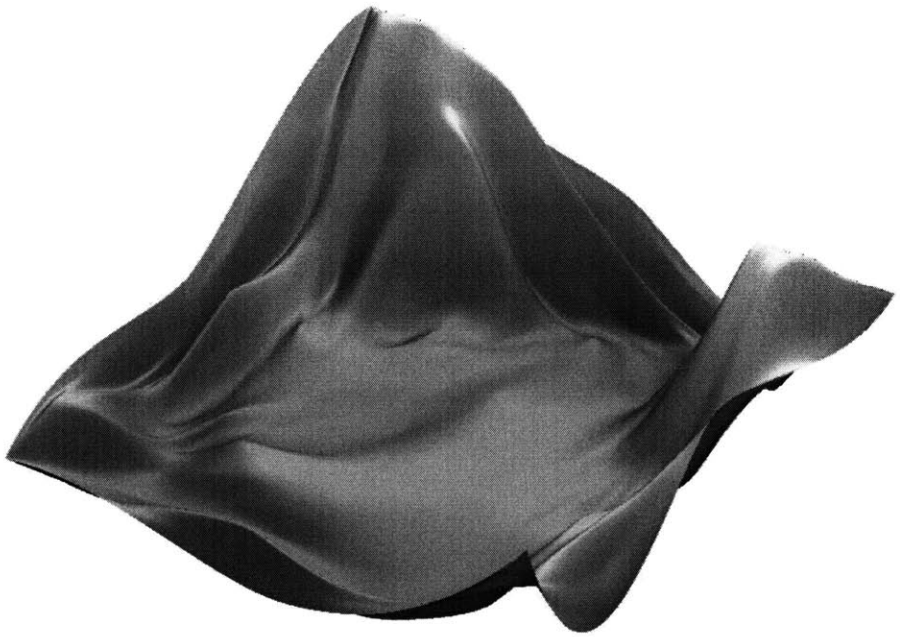
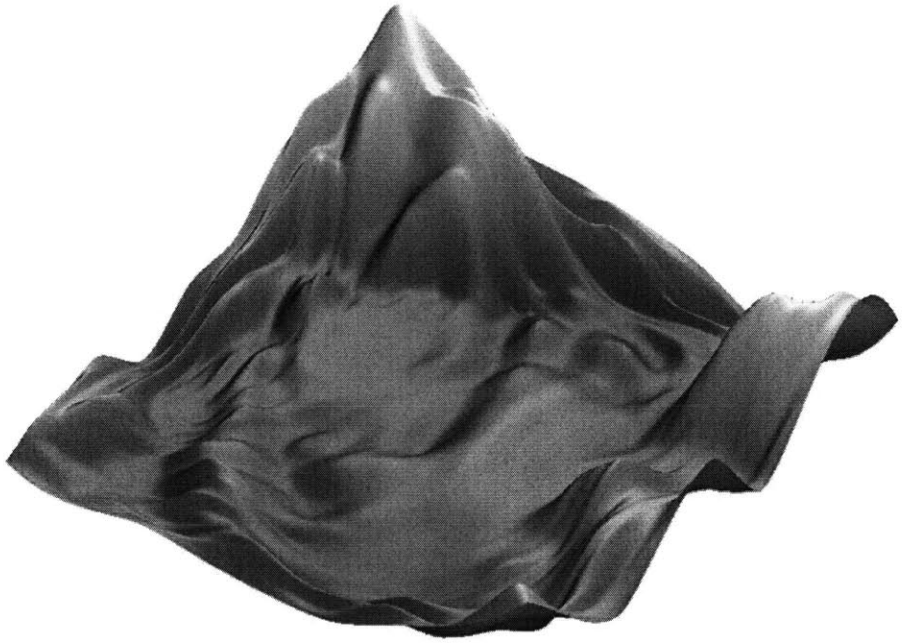


Figure 8-18: WTIN representation with threshold compression at 3% of the height range



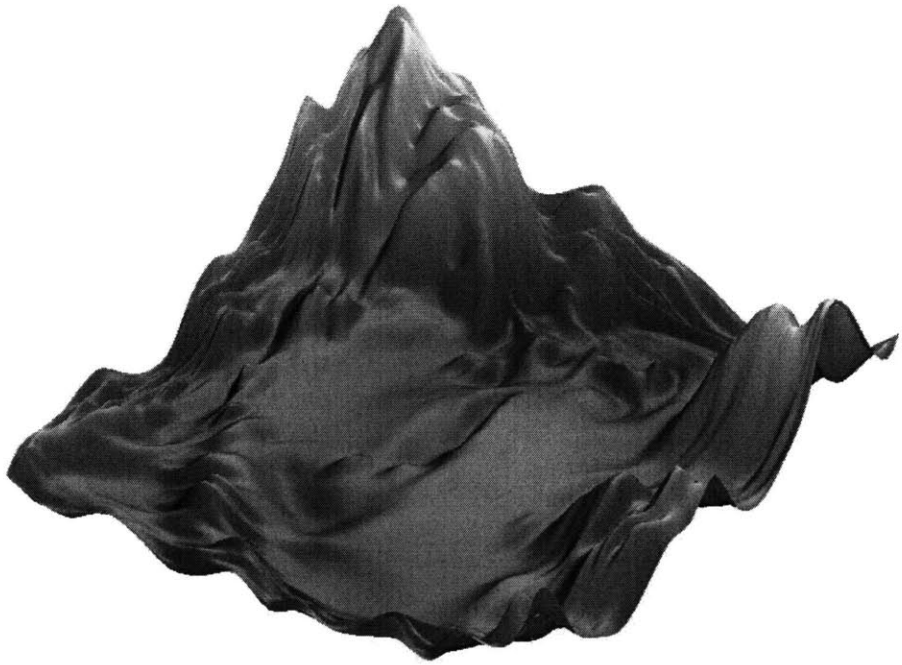
Percentage of wavelet coefficients retained = 0.43%

Figure 8-19: WTIN representation with 2 levels of wavelet coefficients



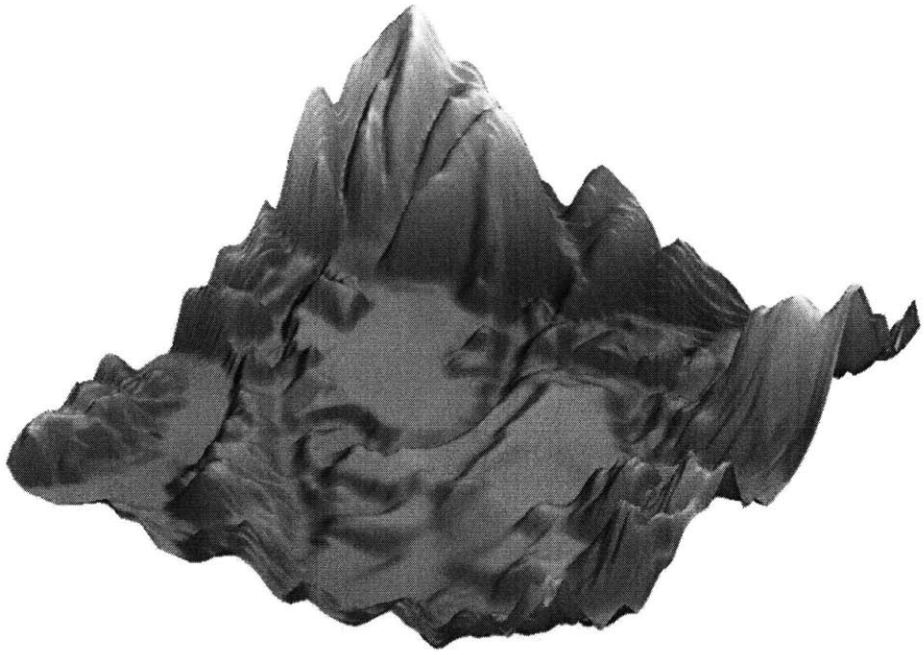
Percentage of wavelet coefficients retained = 1.62%

Figure 8-20: WTIN representation with 3 levels of wavelet coefficients



Percentage of wavelet coefficients retained = 6.35%

Figure 8-21: WTIN representation with 4 levels of wavelet coefficients



Percentage of wavelet coefficients retained = 25.13%

Figure 8-22: WTIN representation with 5 levels of wavelet coefficients

the measure—Peak Signal to Noise Ratio (PSNR), which is defined in equation (8.5).

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{N \cdot H^2}{\sum_1^N (z - \tilde{z})^2} \right) = 10 \cdot \log_{10} \left(\frac{H^2}{\text{mean squared error}} \right) \text{ dB} . \qquad (8.5)$$

Here, N is the number of points, H is the height range (max height—min height), z is the value of the variable of interest and $\tilde{z}$ is the value reconstructed after compression. The unit dB, i.e. decibel, is a non-dimensional unit. It is used to express PSNR in a logarithmic scale. For example, a PSNR of 30 dB corresponds to a mean squared error of $H^2/1000$. Notice that this PSNR differs from the one we used in Chapter 4 in that the range value H in equation (8.5) varies with the data set, while for gray images, we always use 255 as H to compute the PSNR. The PSNR versus threshold plot is shown in Figure 8-23. The plot of compression ratio versus threshold is shown in Figure 8-24. Here the compression ratio is the total number of wavelet coefficients divided by the number of wavelet coefficients left after the thresholding operation.

Figures 8-23 and 8-24 show that a 5% threshold applied to all seven levels results in a PSNR of 30dB and a compression ratio of 25:1. Figure 8-25 gives the plot of PSNR versus the number of levels of wavelet coefficients included. A number 5 on the horizontal axis means that the first five levels of wavelet coefficients are included in the synthesis transform. Figure 8-26 presents the result of applying the 5% threshold operation on the wavelet coefficients used in Figure 8-25.

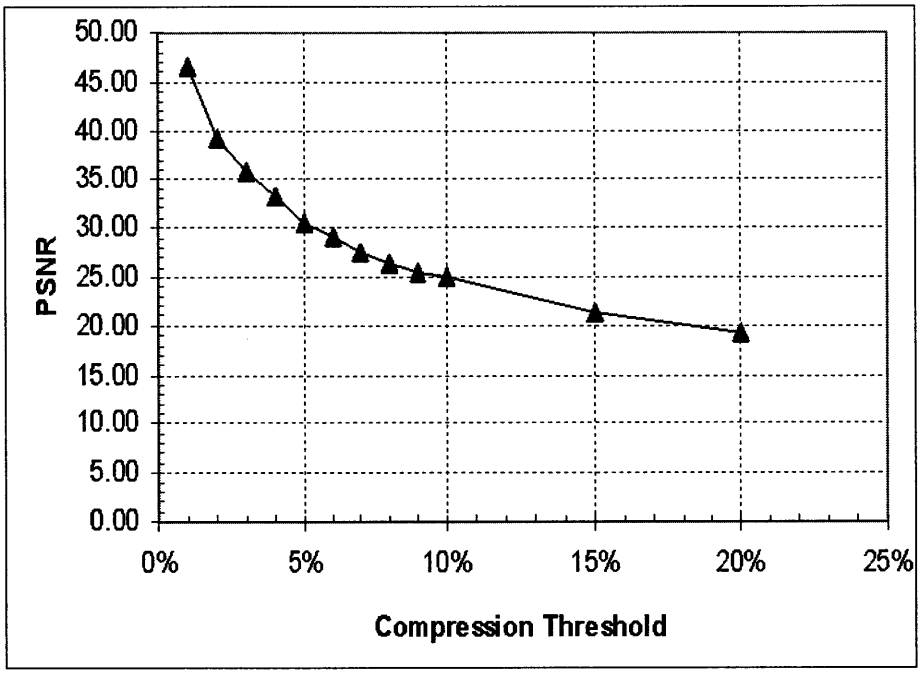


Figure 8-23: Compression: PSNR vs. threshold

The above results show that the WTIN format is an efficient way to represent terrain surfaces. In order to better understanding the properties of a WTIN, we have done some comparisons with other approaches to decompose a terrain surface into a multiresolution format. One possible approach is to obtain the subdivision points in the $x - y$ plane using a simple linear subdivision on the x and y coordinates of the initial configuration, i.e. the middle points of the edges. After that, the height field is resampled on these subdivision points. In the analysis transform, a modified Butterfly scheme can be applied solely on the z coordinates. This process really gives a *hybrid* wavelet transform. However, this hybrid transform creates some visual

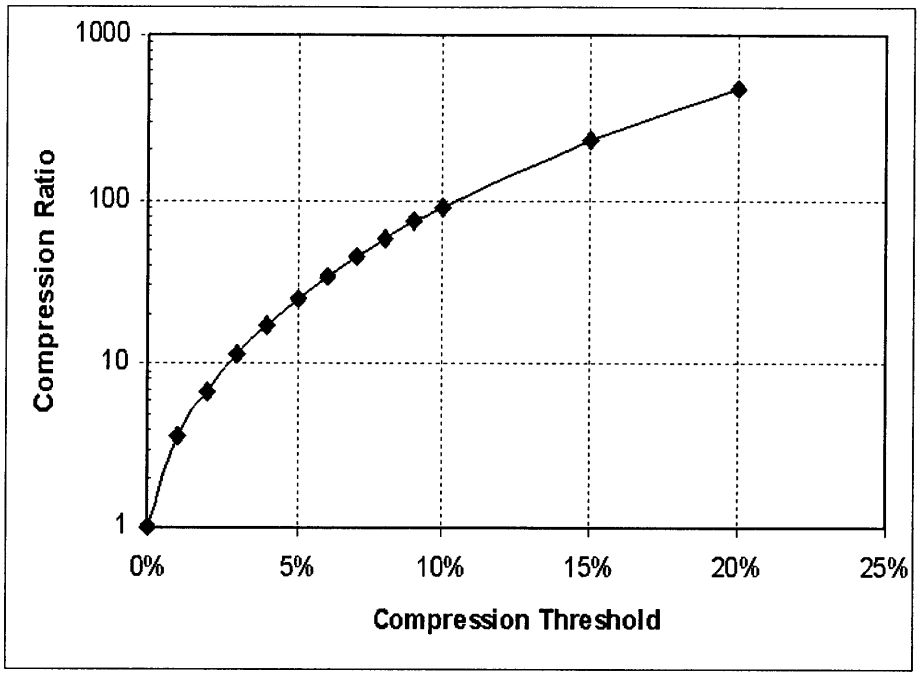


Figure 8-24: Compression: compression ratio vs. threshold

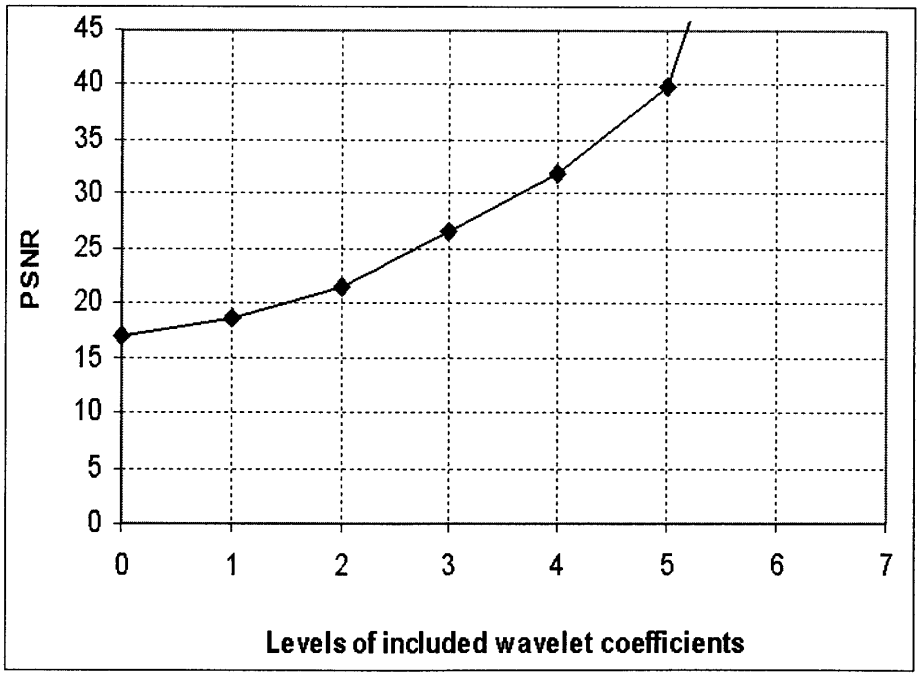


Figure 8-25: Compression: PSNR vs. levels of wavelet coefficients included

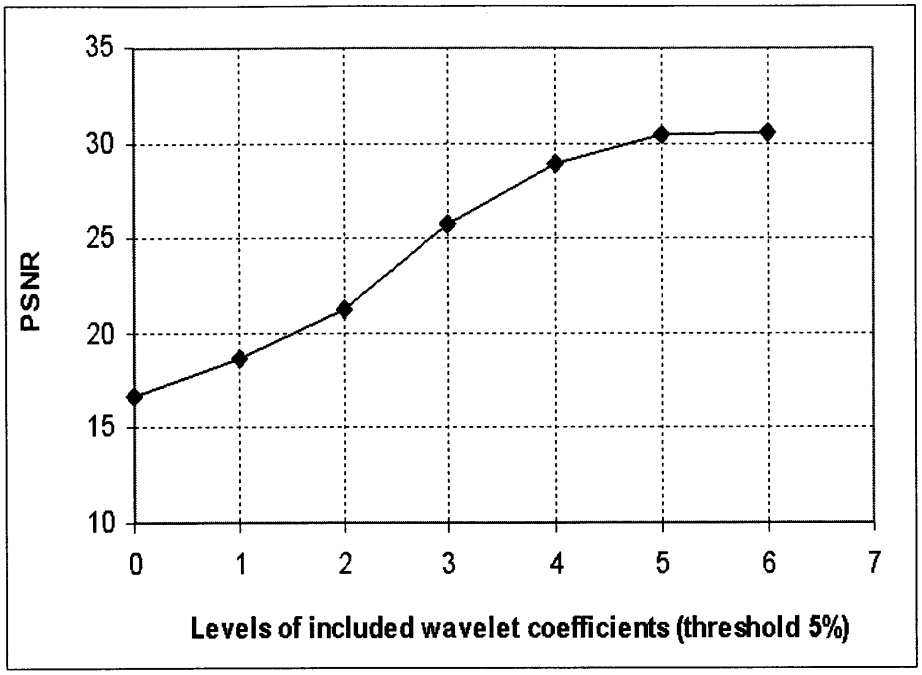


Figure 8-26: Compression: PSNR vs. levels and threshold (5%)

aberrations associated with the regularity of mid-point subdivision. Figure 8-27 shows the result. Given the simplicity of this hybrid algorithm, it may be useful in some GIS applications if these visual aberrations are not harmful to the applications. Another comparison is to compare our result with the result by using the surface Hat wavelet, which is a pure linear filter (linear on three directions x, y, and z). Figure 8-28 tells us that a WTIN based on our interpolation wavelet transform is much better than using the Hat wavelet because it produces much less sharp changes. Finally, an example of overlapping a WTIN with the corresponding image map is shown in Figure 8-29. The data are from the ArcView sample data sets by courtesy of ESRI.

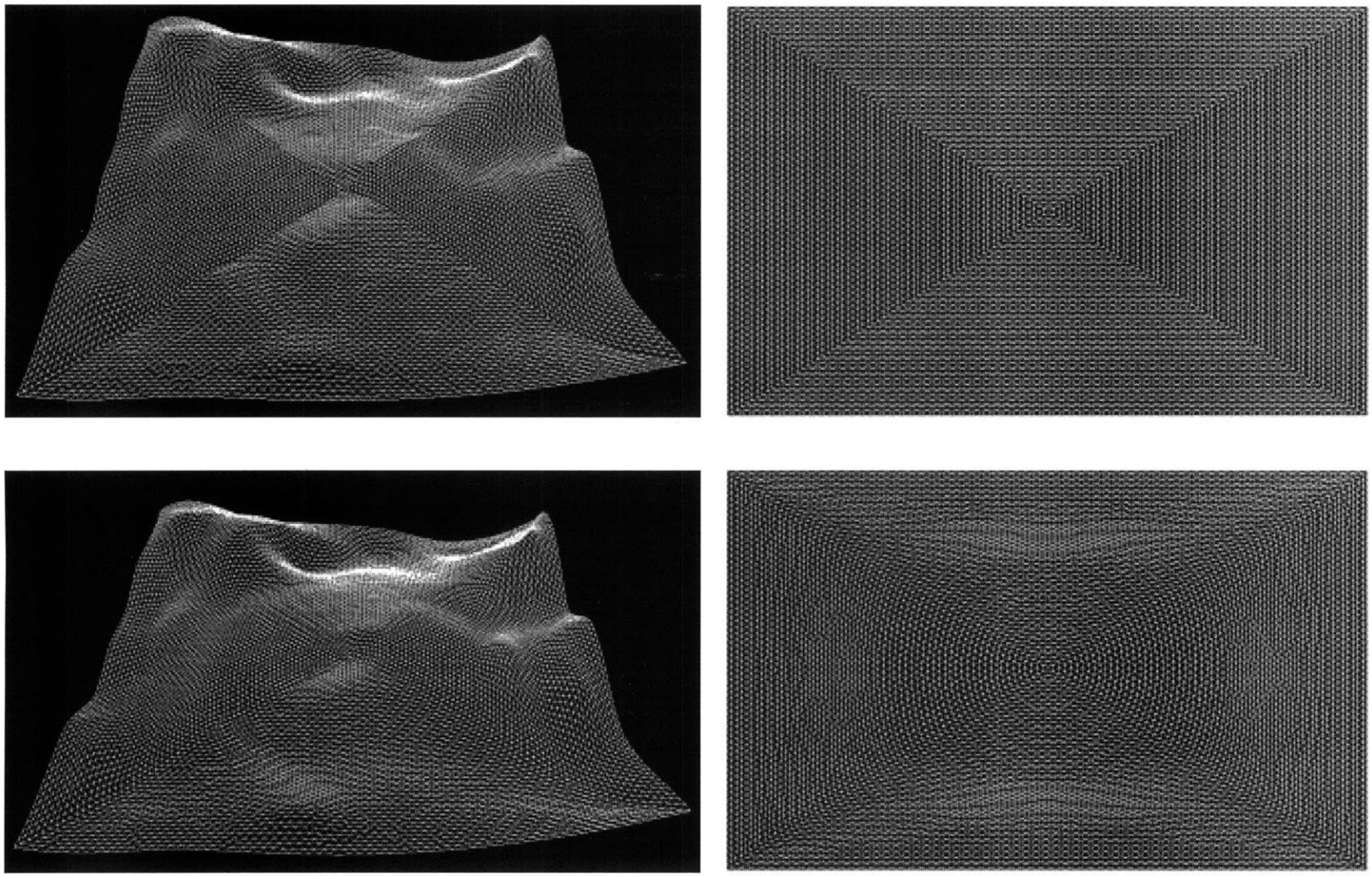
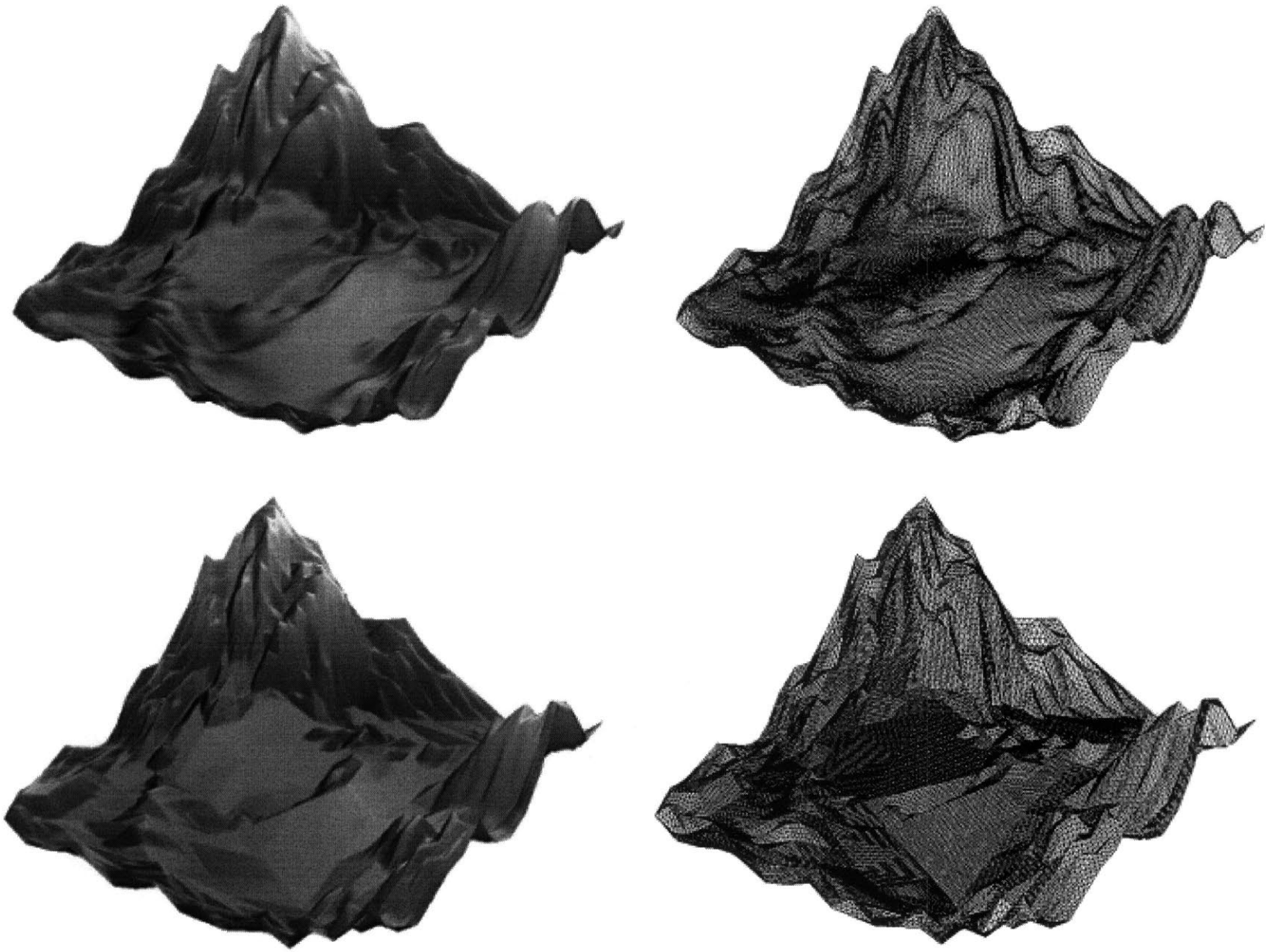


Figure 8-27: Comparison of a WTIN with the result from linear hybrid subdivision (top 2 plots correspond to linear hybrid subdivision, bottom 2 plots correspond to WTIN)



Both examples include 4 levels of wavelet coefficients and 5% threshold compression

Figure 8-28: Comparison of using the proposed modified Butterfly wavelet with the Hat (linear) wavelet (top 2 plots correspond to proposed wavelet, bottom 2 plots correspond to Hat wavelet)

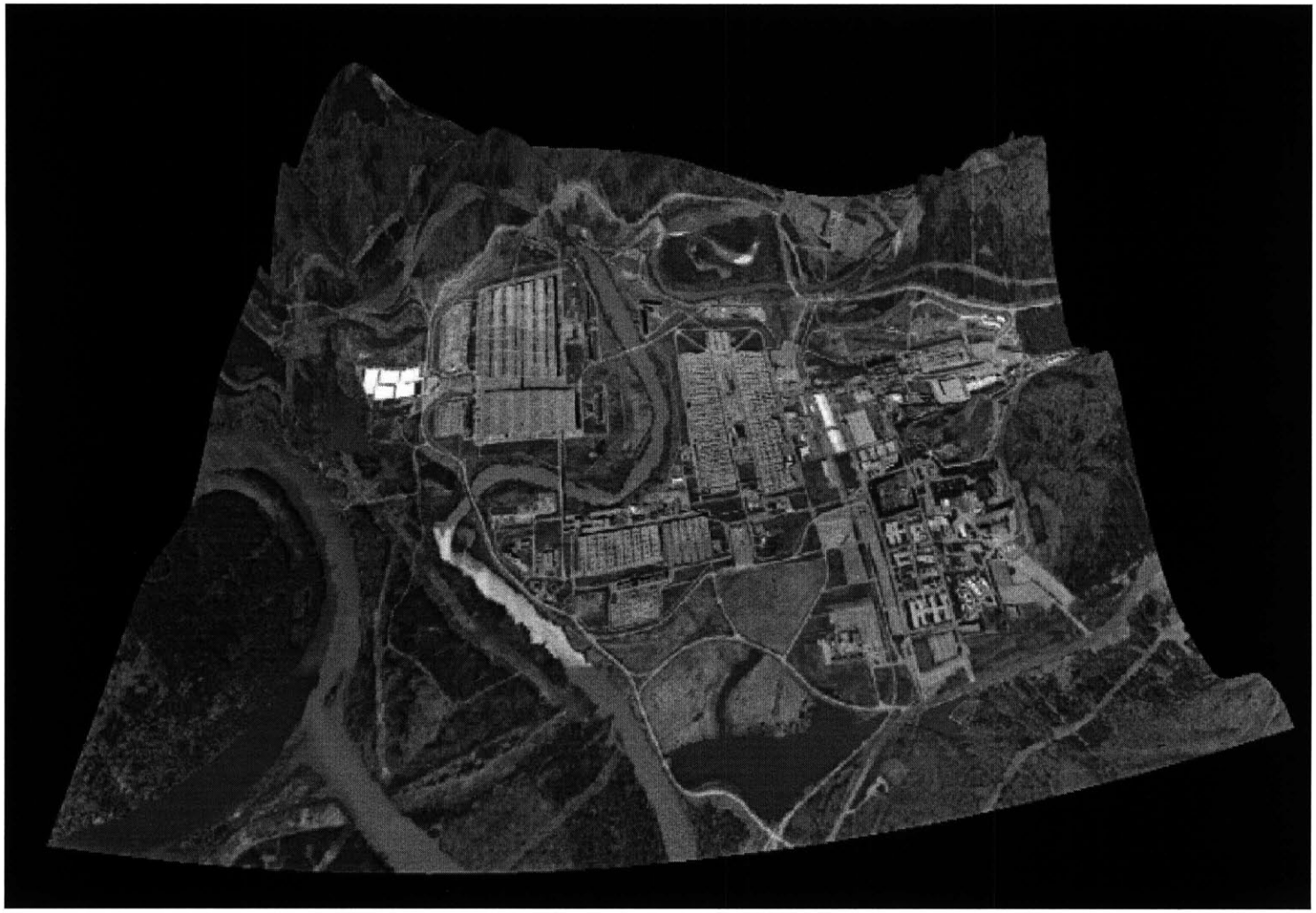


Figure 8-29: Overlap of WTIN and the corresponding image map

Chapter 9

Conclusions

In this chapter, we summarize the main contributions of this research and then offer suggestions for future research.

9.1 Contributions

We have investigated the applications of wavelet theory in scalable distributed GIS services. With the rapid development of the Internet, traditional desktop computing has begun to move to online computing, which leads to a revolutionary change of software systems. Because of the scalability and bandwidth limitations of networks, there has been a need for new research to solve the new problems of online computing. In the GIS domain, since more data and services are expected to be online, good multiresolution data representations are badly needed for business and management. Our work has provided a suite of multiresolution data representations for GIS data based on wavelet theory. We have designed multiresolution data representations for the most common data formats in current GISs. These include WImg for image maps, WDEM for DEM data, WArc for curves, and WTIN for triangulated networks. These representations have the following advantages compared to traditional data formats and other multiresolution formats:

- A solid mathematical foundation from wavelet and subdivision theories;

- Multiresolution capability, suitable for different scalabilities, such as bandwidth, device, and demand scalability;

- Fast transforms, since the filtering is a recursively linear operation (only constant matrix multiplications are involved, excluding the thresholding operation) and has local support;

- Easy to compress due to the large number of wavelet coefficients with small magnitudes.

In designing the WImg and WDEM formats, we have used the first generation wavelet theory. Besides the general compression capability of wavelet transforms that has been explored by other researchers, we have demonstrated the dynamic zooming and panning capabilities of using WImg in a GIS. This is based on the lifting scheme, integer transforms, and tile map management by simple indices. We have noticed that MrSID^{TM} from LizardTech, Inc. and ECW from Earth Resource Mapping, Inc. use similar approaches to decompose image maps, which demonstrates that the idea of using wavelets in a GIS is commercializable and promising. We have not found any parallel work for DEM data, however. A comparable format is the raster pyramid format in ArcInfo from ESRI. The raster pyramid format has only a downsampling operation to obtain a reduced version of a DEM. This method is obviously not as good as the WDEM format, which uses wavelet transforms to explore the correlation among data. Furthermore, the raster pyramid format stores redundant data, so it requires extra storage.

We have also investigated the possibility of applying second generation wavelets to process curves. We have proposed the WArc format based on this research. WArc has the capability to simplify curve features in GIS, such as coastlines. It also provides a multiresolution representation of curve features. The numerical experiments in Chapter 7 have demonstrated the compression capability of the WArc format as well. Related research is the work of Finklestein and Salesin [22], which builds a multiresolution analysis for endpoint-interpolating B-spline curves for scan conversion. In scan conversion, the continuity requirement is relaxed. Compared to their work,

our approach based on the lifting scheme is more efficient and can deal with general GIS data, which may be very irregular.

The most important contribution of this research is the development of the WTIN representation. This is also based on the second generation wavelet theory. Through the lifting scheme, we have proposed a method to construct an efficient multiresolution representation of a terrain surface model. Compared to general wavelet techniques developed for 3D object modeling in the computer graphics domain, where three wavelet coefficients are stored for each vertex, our WTIN model only needs to store one wavelet coefficient for each vertex in a terrain surface. This greatly reduces the storage memory and saves bandwidth for online GIS services. This methodology is based on the assumption that terrain surfaces are functions of two variables, which distinguishes our model from a general three dimensional graphics model. The WTIN model is an interpolation scheme, suitable for many GIS applications. We have also investigated the possibility of schemes that are non-interpolating.

We have also demonstrated a method for transforming high resolution DEM data into a WTIN. An algorithm for choosing feature points has been designed to provide the initial configuration of a WTIN. Then a bilinear interpolation method is used to resample the height field. The numerical experiments have shown that the constructed WTIN gives an efficient multiresolution representation of the original terrain surface. The high compression potential is also demonstrated in the discussion of the results from numerical experiments. Compared to pure subdivision schemes, wavelet analysis based on the lifting scheme provides complete information; however, pure subdivision schemes provide only low frequency information without any high frequency information. Compared to other heuristic methods, such as Hoppe's method [30, 29], our approach is more computationally efficient. This opens a new direction for processing GIS terrain data.

In summary, our main contributions are:

- We have provided a complete set of wavelet-based multiresolution data representations (WImg, WDEM, WArc, and WTIN) for scalable distributed GIS services.

- We have developed dynamic zooming and panning functionalities for tiled image maps based on the WImg format.
- The WDEM format provides effective results on using wavelets to decompose DEM data, which is better than the ArcInfo raster pyramid format.
- The WArc format provides curve simplification capability and multiresolution representations for general parameterized curvilinear features.
- The WTIN format uses one wavelet coefficient for each vertex, much simpler than general 3D computer graphics models, while achieving good results.
- We have provided an efficient algorithm to choose the initial configuration from a high resolution DEM. This makes it easy in practice to transform a DEM to a WTIN.

9.2 Future Research Directions

Wavelet theory has been developed for more than two decades. Most research is focused on mathematical theories and first generation wavelets. The success of applying first generation wavelets to image processing has demonstrated the great potential of wavelets in real-life applications. The lifting scheme and the subsequent construction of second generation wavelets have further extended the possibility of using wavelets in broader application areas than those with uniformly spaced data. The concept of multiresolution is more fundamental than wavelet analysis. Wavelet analysis is only one approach to obtain multiresolution. Multiresolution is more philosophic than technical to humans. Therefore, a lot of research opportunities exist to expand this research. Below are some thoughts on future research directions.

1. Currently, we have developed four different multiresolution representations for different data sets. The layering technique is used to integrate them to express richer data sets in GIS. A higher level of integration needs to be further developed to combine different CAD and VRML models to achieve more content-

based simulations. The final system will be merged into a virtual reality (VR) system. A 3D VR navigator will be used in a modern automobile system in the near future, which requires more scalable data than geographical data alone. Many issues may arise in the integration process, such as overall performance tuning and the consistency of map coordinate transformations . All these deserve more research.

2. The proposed WTIN format is still in its infancy and further research opportunities exist to make it more mature. One direction is to combine a WTIN with rendering level techniques to achieve adaptive level-of-detail control. Directly implementing the adaptive feature in a WTIN may not be worthwhile because it will destroy the internal structure and increase the storage and transmission of information, which deviates from the objectives of saving bandwidth and space. Therefore, we propose to separate the rendering layer and the storage and transmission layer, so that each layer yields the best result for its particular focus. There is a considerable amount of research on the topic of rendering level-of-detail control. Hoppe [30] provides a good source for this. Note that even in the current implementation, smooth regions will result in small wavelet coefficients, which will be discarded during the compression operation. Therefore, while adaptive subdivision can improve rendering performance, it is not expected to significantly improve the compression performance. Another direction to improve the WTIN format is to extend it to include an approximation scheme. Currently, the WTIN format is designed as an interpolation scheme. This satisfies many GIS applications. However, some GIS applications may not have this requirement; then an approximation scheme can give a better representation due to the redistribution of wavelet coefficients (the updating step in the lifting scheme). However, currently there is not much theoretical work on how to do this in a general triangular network. The third direction is to investigate carefully designed quantization schemes to replace the currently used simple thresholding quantization and better encode wavelet coefficients for a

WTIN. This will result in better compression.

3. More theoretical work on wavelets for irregular setting data is needed. The current framework of general 3D wavelet transforms and our WTIN model all involve resampling operations. This satisfies most applications' requirements. However, a non-resampling scheme may be more attractive if we do not sacrifice too much on data size. In this way, connectivity information may need to be stored in a very compact form. Some recent research [14, 40] has revealed this is possible. On the other side, the topological surgery method [50] has demonstrated that we can expand a general 3D surface into a 2D spanning tree. If we can build a general wavelet transform on this spanning tree, a non-resampling scheme for compressing 3D geometries will be very promising.

4. Generally speaking, the quality of the initial configuration affects the quality of the final surface. The triangular shape in the initial configuration also affects the boundary networks. Therefore, a good initial configuration is important. One approach is to use edge collapse and vertex merge to gradually reduce raw triangular data. But this approach is computationally expensive. We have developed an algorithm to pick up feature points to construct the initial configuration of a WTIN. However, because it is not the major focus of our research, we have not pursued this in great depth at this point. Therefore, further research in this direction is desirable.

5. There are opportunities to build simpler and more efficient subdivision algorithms for triangular networks. Figure 9-1 shows a preliminary work on this. The left plot shows a shrinking subdivision. For every new subdivision point, such as A, B, or C in Figure 9-1, six points in the coarse resolution, $\{K_i\}$, are involved in the evaluation process. The limiting case is shown on the right plot, where every triangular face in a coarser level shrinks to a point. The limiting case is similar to the $\sqrt{3}$ subdivision algorithm proposed by Kobbelt [32]. The new subdivision scheme can construct complete second order parameter polynomial surfaces $\{1, s, t, s^2, t^2, st\}$, which may achieve the same continuity

as the Butterfly scheme, but uses fewer points to evaluate new points, which may result in a more efficient algorithm.

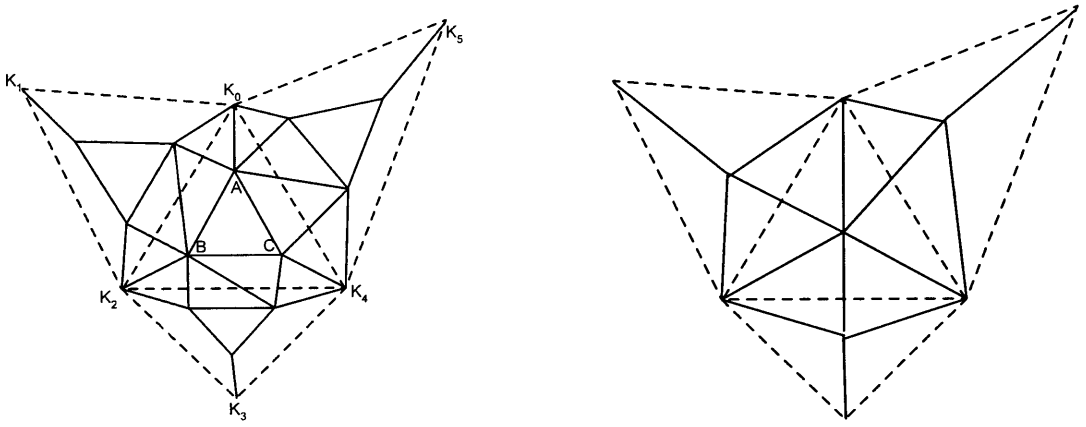


Figure 9-1: Shrinking subdivision scheme (dashed lines are coarser representations)

6. A good extension of this work is to combine the WTIN model with physical numerical simulation, such as pollution distribution problems or finite element simulations. Since multiscale simulation is promising for fast computation, wavelets may play an important role as a hierarchical basis. The WTIN model is a good candidate data representation for geographical related computation as well.

7. Generally, we design a second generation wavelet based on a better polynomial accuracy criterion. However, this does not utilize the characteristics of human visual system. Therefore, other criteria can be investigated to improve the overall wavelet transform quality in processing visualization data. Optimization in the L^2 norm is one possible criterion.

Above are some of the possible extensions to our current research. Multiresolution representations and wavelet transforms are powerful concepts and tools. We believe that they will continue to play a critical role in technology innovations.

Bibliography

[1] Mahdi Abdelguerfi, Chris Wynne, Edgar Cooper, and Ladner Roy. Representation of 3-d elevation in terrain databases using hierarchical triangulated irregular networks: a comparative analysis. *International Journal of Geographic Information Science*, 12(8):853–873, 1998.

[2] Chandrajit L. Bajaj, Valerio Pascucci, and Guozhong Zhuang. Progressive compression and transmission of arbitrary triangular meshes. In *Proceedings of IEEE Visualization Conference*, pages 307–316, Oct 1999.

[3] Michela Bertolotto and Max J. Egenhofer. Progressive vector transmission. In *Proceedings of 7th ACM Symposium on Advances in Geographical Information Systems*, pages 152–157, Nov 1999.

[4] Barbara P. Buttenfield. Progressive transmission of vector data on the Internet: a cartographic solution. In *18th International Cartographic Conference Proceedings*, Ottawa, Canada, August 1999.

[5] Charles K. Chui. *An Introduction to Wavelets.* Academic Press, 1992.

[6] James H. Clark. Hierarchical geometric models for visible surface algorithms. *Communications of ACM*, 19(10):547–554, October 1976.

[7] A. Cohen, I. Daubechies, and J.-C. Feauveau. Biorthogonal bases of compactly supported wavelets. *Communication on Pure and Applied Mathematics*, 45:485–560, 1992.

[8] J. T. Coppock and D. W. Rhind. The history of GIS. In D. J. Maguire, M. Goodchild, and D. W. Rhind, editors, *Geographic Information Systems: Principles and Applications*, pages 21–41. Longman, New York, 1991.

[9] I. Daubechies, I. Guskov, P. Schröder, and W. Sweldens. Wavelets on irregular point sets. *Phil. Trans. R. Soc. Lond. A*, 357(1760):2397–2413, 1999.

[10] Ingrid Daubechies. Orthonormal bases of compactly supported wavelets. *Comm. Pure and Appl. Math.*, 41:909–996, 1988.

[11] Ingrid Daubechies. *Ten Lectures on Wavelets.* SIAM, Philadelphia, PA, 1992.

[12] Leila De Floriani. A pyramidal data structure for triangle-based-surface description. *IEEE Computer Graphics and Applications*, 9(2):67–78, 1989.

[13] Leila De Floriani, Paola Magillo, and Enrico Puppo. Compressing TINs. In *Proceedings of 6th ACM Workshop on Advances in Geographical Information Systems*, pages 130–145, Washington DC, November 1998.

[14] Leila De Floriani, Paola Magillo, and Enrico Puppo. Compressing triangulated irregular networks. *GeoInformatica*, 4(1):67–88, 2000.

[15] Leila De Floriani and Enrico Puppo. A hierarchical triangle-based model for terrain description. In A. U. Frank, I. Campari, and U. Formentini, editors, *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space*, number 639 in Lecture Notes in Computer Science, pages 236–251. Springer-Verlag, New York, 1992.

[16] Leila De Floriani and Enrico Puppo. Hierarchical triangulation for multiresolution surface description. *ACM Transaction on Graphics*, 14(4):363–411, October 1995.

[17] D. L. Donoho. Unconditional bases are optimal bases for data compression and for statistical estimation. *Applied Computational Harmonic Analysis*, 1(1):100–115, December 1993.

[18] David H. Douglas and Thomas K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *The Canadian Cartographer*, 10(2):112–122, December 1973.

[19] N. Dyn, D. Levin, and J. A. Gregory. A butterfly subdivision scheme for surface interpolation with tension control. *ACM Transaction on Graphics*, 9(2):160–169, June 1990.

[20] M. Eck, T. Derose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle. Multiresolution analysis of arbitrary meshes. In *ACM SIGGRAPH 95 Proceedings*, pages 173–182, 1995.

[21] Gerald Farin. *Curves and Surfaces for Computer Aided Geometric Design: a Practical Guide.* Computer Science and Scientific Computing. Academic Press, Chestnut, MA, fourth edition, 1997.

[22] A. Finkelstein and D. H. Salsin. Multiresolution curves. In *(ACM SIGGRAPH 94) Proceedings*, pages 261–268, New York, NY, USA, 1994.

[23] M. Garland and P. S. Heckbert. Fast polygonal approximation of terrains and height fields. Technical report, CS Department, Carnegie Mellon University, Pittsburgh, PA, USA, 1995.

[24] J. D. Gibson, T. Berger, T. Lookabaugh, D. Lindbergh, and R. L. Baker. *Digital Compression for Multimedia Principles and Standards.* Morgan Kaufmann, San Francisco, 1998.

[25] C. M. Gold and S. Cormack. Spatially ordered networks and topographic reconstruction. *International Journal of Geographic Information Science*, 1:137–148, 1987.

[26] A. Grossmann and J. Morlet. Decomposition of Hardy functions into square integrable wavelets of constant shape. *SIAM J. Math. Anal.*, 15:723–726, 1984.

[27] P. S. Heckbert and M. Garland. Survey of polygonal surface simplification algorithms. Technical report, Carnegie Mellon University, Pittsburgh, PA, USA, 1997.

[28] Hugues Hoppe. Progressive meshes. In *ACM SIGGRAPH 96 proceedings*, pages 99–108, 1996.

[29] Hugues Hoppe. Efficient implementation of progressive meshes. *Computers & Graphics*, 22(1):27–36, 1998.

[30] Hugues Hoppe. Smooth view-dependent level-of-detail control and its application to terrain rendering. In *IEEE Visualization 98 Conference Proceedings*, pages 35–42, Research Triangle Park, NC, USA, October 1998.

[31] Bernd Jünger and Jack Snoeyink. Selecting independent sets for terrain simplification. In *Proceedings of WSCG '98*, pages 157–164, Plzen, CZ, February 1998.

[32] Leif Kobbelt. $\sqrt{3}$-subdivision. In *ACM SIGGRAPH 00 proceedings*, pages 103–112, 2000.

[33] A. W. F. Lee, W. Sweldens, P. Schröder, L. Cowsar, and D. Dobkin. MAPS: Multiresolution Adaptive Parameterization of Surfaces. In *ACM SIGGRAPH 98 proceedings*, pages 95–104, 1998.

[34] Charles Teorell Loop. Smooth subdivision surfaces based on triangles. Master's thesis, University of Utah, 1987. Department of Mathematics.

[35] M. Lounsbery. *Multiresolution Analysis for Surface of Arbitrary Topological Type.* PhD thesis, University of Washington, Seattle, WA, USA, 1994.

[36] M. Lounsbery, T. D. DeRose, and J. Warren. Multiresolution surfaces of arbitrary topological type. *ACM Transaction on Graphics*, 16(1):34–73, 1997.

[37] Stephane G. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 11(7):674–693, July 1989.

[38] Y. Meyer. *Wavelets, Algorithms and Applications.* SIAM, Philadelphia, PA, USA, 1993.

[39] Alan V. Oppenheim, Ronald W. Schafer, and John R. Buck. *Discrete-Time Signal Processing.* Prentice Hall Signal Processing Series. Prentice-Hall Inc., Upper Saddle River, New Jersey, second edition, 1999.

[40] Donggyu Park, Hwangue Cho, and Yangsoo Kim. A TIN compression method using Delaunay triangulation. *International Journal of Geographic Information Science*, 15(3):255–269, 2001.

[41] Johnathan F. Raper. The 3-dimensional geoscientific mapping and modeling system: a conceptual design. In Johnathan F. Raper, editor, *Three Dimensional Applications in GIS*, page 11. Talor & Francis Ltd, Philadelphia, PA, USA, 1989.

[42] Lori Scarlatos and Theo Pavlidis. Hierarchical triangulation using cartographic coherence. *CVGIP: Graphical Models and Image Processing*, 54(2):147–161, March 1992.

[43] Peter Schröder and Wim Sweldens. Spherical wavelets: Efficiently representing functions on the sphere. *ACM SIGGRAPH 95 Proceedings*, pages 161–172, 1995.

[44] Peter Schröder and D. Zorin. Subdivision for modeling and animation. In *SIGGRAPH 98 Course Notes.* ACM SIGGRAPH, 1998.

[45] Jack Snoeyink and M. van Kreveld. Liner-time reconstruction of Delaunay triangulator. In *Proceedings of 5th European Symposium on Algorithms*, pages 459–471, New York, 1997. Springer-Verlag.

[46] Eric J. Stollnitz, Tony D. DeRose, and David H. Salesin. *Wavelets for Computer Graphics Theory and Applications.* Morgan Kaufmann Publishers, Inc., San Francisco, California, 1996.

[47] Gilbert Strang and Truong Nguyen. *Wavelets and Filter Banks.* Wellesley-Cambridge Press, Wellesley, MA, 1996.

[48] Wim Sweldens. The lifting scheme: A custom-design construction of biorthogonal wavelets. *Appl. Comput. Harmon. Anal.*, 3(2):186–200, 1996.

[49] Wim Sweldens. The lifting scheme: A construction of second generation wavelets. *SIAM Journal on Mathematical Analysis*, 29(2):511–546, 1997.

[50] Gabriel Taubin and Jarek Rossignac. Geometric compression through topological surgery. *ACM Transactions on Graphics*, 17, 1998.

[51] Martin Vetterli and Kovačević. *Wavelets and Subband Coding.* Prentice Hall PTR, Upper Saddle River, New Jersey, 1995.

[52] Michael Zeiler. *Modeling Our World The ESRI Guide to Geodatabase Design.* Environmental Systems Research Institute, Inc., Redlands, CA, USA, 1999.

[53] Denis N. Zorin. Interpolating subdivision for meshes with arbitrary topology. In *ACM SIGGRAPH 96 Proceedings*, pages 189–192, New Orleans, Louisiana, USA, August 1996.

[54] Denis N. Zorin. *Stationary Subdivision and Multiresolution Surface Representation.* PhD thesis, California Institute of Technology, Pasadena, CA, USA, 1998.

[55] Denis N. Zorin. Subdivision zoo. In *Subdivision for Modeling and Animation*, SIGGRAPH 2000 Course Notes. ACM SIGGRAPH, 2000.